INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

Statistical Machine Translation - The Problem of Unknown Words

João Pedro Vasques Filipe da Silva

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática e de Computadores

Júri

Orientador:	Professor Luís Eduardo Teixeira Rodrigues
Orientador:	Professora Maria Luísa Torres Ribeiro Marques da Silva Coheur
Co-orientador:	Professora Isabel Maria Martins Trancoso
Vogal:	Professor Bruno Emanuel de Graça Martins

Maio 2012

Agradecimentos

Um grande obrigado à Professora Maria Luísa Coheur, pelo seu apoio incondicional durante todo este processo, por me dar sempre os melhores conselhos e por mostrar um grande entusiasmo pelo projecto, concedendo-me a motivação necessária para dar o meu melhor. Um obrigado também à Professora Isabel Trancoso, pelas suas sempre sábias sugestões.

Agradeço a todos os meus amigos, que me ajudaram a ultrapassar os momentos mais delicados e que por me pedirem para explicar em que consistia a minha tese, fizeram com que inadvertidamente eu a percebesse melhor. Um obrigado especial à minha namorada Kiran e ao meu grande amigo Oliver, que mesmo não estando constantemente presentes, fizeram tudo o que podiam para me ajudar e motivar.

Um enorme obrigado a toda a minha família, por estarem ao meu lado a apoiar-me em todos os passos que dou e por estarem prontos a ajudar em qualquer ocasião. O meu maior agradecimento vai para a minha mãe Ana, sem a qual nunca conseguiria ter chegado onde estou hoje, tanto pessoal como profissionalmente. Este trabalho é dedicado a ela.

Obrigado a todos por sempre acreditarem em mim.

Acknowledgements

Thank you to Professor Maria Luísa Coheur, for her unconditional support throughout this whole process, for always giving me the best advice and for showing constant enthusiasm for this project, granting me the motivation I needed to give my best. Thank you also to Professor Isabel Trancoso, for her always wise suggestions.

To all my friends, who helped me go through the most delicate moments and who, by asking me to explain my thesis, inadvertently helped me to understand it better. A special thank you to my girlfriend Kiran and to my good friend Oliver, who, even not being able to be present at all times, did all they possibly could to help me and to motivate me.

A big thank you to my whole family, for being by my side, supporting me in every step I take and for being ready to help in any occasion. My biggest thank you goes to my mother Ana, without whom I would have never been able to be where I am today, both personally and professionally. This work is dedicated to her.

Thank you all for always believing in me.

Abstract

Developing systems that translate words as accurately as humans is not an easy task. Statistical Machine Translation systems base themselves on training data. So, when translating a document, some of the words in that document might not have been encountered in the training phase and, thus, the system does not know how to translate these words. The objective of this work is to develop a system that finds possible translations of these unknown words.

Since words from closely related languages have common ancestors, many of these words will end up having similarities that can help us in discovering if they are possible translations of each other or not, these words are named cognate words. Therefore, we explore orthographic similarities between words to find translations.

We also make use of Logical Analogy, by attempting to infer the translation of the unknown word by looking at the translations of words related to it.

Our final method tested uses the context in which each word is inserted to calculate how similar two words are.

By merging these systems, we try to maximize the number of unknown words translated. Our approach is tested in the translation of corpora from Portuguese to English (and vice-versa).

Resumo

Desenvolver sistemas que traduzem palavras com a mesma exatidão que humanos não é uma tarefa fácil. Sistemas de Tradução Automática Estatística baseiam-se no treino de data. Logo, quando traduzem um documento, certas palavras do documento podem não ter sido encontradas na fase de treino e, por conseguinte, o sistema não sabe como as traduzir. O objectivo deste trabalho é desenvolver um sistema que encontra possíveis traduções destas palavras desconhecidas.

Dado que palavras em línguas com a mesma origem têm antepassados em comum, muitas destas palavras têm semelhanças que podem ser úteis para descobrir se estas são ou não tradução uma da outra, tais palavras são denominadas cognatas. Por esse motivo, exploramos semelhanças na ortografia de palavras para encontrar traduções.

Neste sistema, também fazemos uso de Analogias, tentando inferir a tradução de palavras através da tradução de palavras relacionadas com ela.

O último método testado usa o contexto em que cada palavra está inserida para calcular o quão similares são duas palavras.

Ao juntar estes módulos, tentamos maximizar o número de palavras desconhecidas traduzidas. A nossa abordagem é testada na tradução de corpora de Português para Inglês (e vice-versa).

Keywords

- Statistical Machine Translation
- Unknown Words
- Cognates
- Analogy
- Context

Palavras Chave

- Tradução Automática Estatística
- Palavras Desconhecidas
- Cognatas
- Analogia
- Contexto

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Statistical Machine Translation systems base their performance in the possibility of finding high frequent patterns of co-occurrences of words. Therefore, infrequent words have a higher probability of being incorrectly translated. However, in many European Languages, many words have a similar surface form or, at least, sound similar. Moreover, sometimes there are affixes that allow the direct translation of these words (Koehn and Knight [2002]). For instance, in English, the suffix *tion* corresponds to the suffix *ção* in Portuguese, as showed, for instance, in the pair (*intuition*, *intuição*). By the same token, prefixes *hyper* in English and *hiper* in Portuguese match for the same words quite often, as stated by the pair (*hyperactive*, *hiperactivo*). Words that can be translated with one of these strategies are usually **cognates**, that is, words that share the same root, or as it is said by linguists, have a common etymological origin.

Nevertheless, the percentage of cognates between two languages can be low, and other ways to find the translation of unknown words need to be envisaged. Just as there are regular affixes that allow the translation of words between languages, there are also certain analogies between words that can help the translation process of unknown words. As an example, the gerund in English can be obtained by adding *ing* to the end of a verb, as seen in the pairs (*eat*, *eating*) or (*read*, *reading*). Thus, if we manage to translate some of these elements to another language, *e.g.* Portuguese, we can infer the translation of the remaining elements. For instance, if we know the translations (*eat*, *comer*), (*eating*, *comendo*) and (*read*, *ler*) we can infer the translation of *reading* as *lendo*. **Logical Analogy** (Langlais and Patry [2007]) is the technique that allows us to establish those inferences and one of the main focus of this project.

Various other ways to find translations of unknown words exist. For instance, using the **context** of each word (Fung and Yee [1998]), that is, the words that precede and succeed the unknown word. As an example, the word “driving” might be close to the word “car”, just as the word “conduzir” is close to the word “carro”. Using these characteristics we can assume that two words with similar contexts are related somehow.

In this project we attempt to mix different methods that allow us to translate unknown words. If we use each method’s advantages to compensate for another method’s flaws, we can improve the results of the system as a whole and offer better translations for the unknown words.

1.2 Objectives

The objective of this work is to find translation of words that a Statistical Machine Translation (SMT) cannot translate. To achieve this we:

- Search for related work on the subject;
- Implement improvements in a system that detects cognates and translates Named Entities (NE) ([Carvalho, 2010]);
- Introduce other ways to discover the translation of an unknown word, by using analogy with other words or by comparing the context of words;
- Evaluate the proposed system.

1.3 Dissertation Structure

This dissertation is divided into the following chapters:

- Chapter 2 (Related Work) presents previous work dealing with unknown words, divided into works that use cognates, works that use other techniques (including the use of analogy or context, for example) and works that deal with the translation of Named Entities;
- Chapter 3 (Baseline System) introduces the system that this dissertation intends to improve;
- Chapter 4 (Towards Unknown Words Translation) explains how each of the modules inside the system work and how to merge the information received from the different modules;
- Chapter 5 (Evaluation) presents experimental results obtained from running tests on the developed system;
- Chapter 6 (Conclusion and Future Work) draws conclusions on the work developed and considers possible future work for further improvements.

2 Related Work

There are various different methods to deal with unknown words and with the ways on how to translate them. The one focused on the previously mentioned system is cognate detection. A number of other authors also use cognate detection in their systems, some of the works on the subject are shown in Section 2.1. Other techniques apart from cognate detection are explored in Section 2.2. Finally Section 2.3 takes a look at different techniques used for NE translation.

2.1 Cognates

Cognates are words in different languages that have the same origin, for example *night*, *nacht* and *noite* all come from a common root word. Since cognates originate from the same ancestor, they normally have similar spellings, only evolving slightly across times. This property of cognates can be very useful when trying to find words that are cognates. However, there are also words commonly known as *false cognates*, to this example are the verb *realize* in English and the verb *realizar* in Portuguese, these words have the same etymological origin but mean different things. It is impossible to distinguish between these *false cognates* and the *real cognates* we want to obtain (those with same origin and same meaning). Nevertheless, the number of *false cognates* is very small in comparison to *real cognates* so building an SMT that recognizes cognates is viable.

[Kondrak et al., 2003] gives a simple approach into cognate detection by using some simple coefficients that look only at how the words are spelled. The Dice coefficient [see Dice, 1945] looks at how many bigrams the words have in common in comparison to the total number of bigrams in both words. The Longest Common Subsequence Ratio (LCSR) checks how many letters there are in common divided by the number of letters in the longest word. Another approach is shown by [Simard et al., 1992] who considers two words as being cognates if the first 4 letters are the same in both words. Both are very simple methods that do not produce perfect results, but are a good starting point for finding cognates in texts.

There are common changes in words that can be seen when translating from a source language to a target language. One example, given by [Mulloni and Pekar, 2006], is replacing all "*c*"s by "*k*"s when translating from English to German. Ex: *tractor* to *traktor*. [Mulloni and Pekar, 2006] call these regularities *orthographic cues*.

Instead of finding all of these cues manually and introducing them into a translation system, these authors develop a learning algorithm where, given a list of cognate pairs between English-German, it will extract the most common orthographic changes. This can be achieved by going through each cognate pair and determining what edit operations (remove, add or replace letters) need to be made in the English word to reach the German word. For example:

- *toilet/toilette*, add the rule to insert *te* at the end of words;
- *tractor/traktor*, add the rule to replace the letter *c* by the letter *k*;
- *absolute/absolut*, add the rule to remove the letter *e* from the end of words.

At the end of the training phase, each rule is attributed a score to determine a rank of the most common cues. The test mode compares two words to see if they are cognates of each other by calculating a normalized edit distance (NED). If before calculating this NED the most common orthographic changes verified by the learning algorithm are applied, the NED will return a much lower value and calculate with more accuracy the words that are cognates. This can be applied to English/Portuguese as well since there are many patterns that can be found in translations between these languages such as replacing *tion* by *ção*.

A system that detects if two words are cognates can then be used for other things. One example is translating NEs, which will be explained later. [Rama and Borin, 2011] show another application to use a cognate detection system. In this paper the authors use cognates to study how closely related some European Languages are. They extracted all the corpora from Europarl [1] in the available languages: Danish, Dutch, English, Finnish, French, German, Greek, Italian, Portuguese, Spanish and Swedish, but decided to exclude Greek since it is not written with the same characters as the other languages. With this study they wanted to show that by extracting phrase pairs from the Europarl parallel corpora and checking how many of the words in those phrases are cognates, they could determine which languages are closely related and which are not.

To check for cognates, they used Dice and LCSR, two measures mentioned before, but also the Levenshtein Distance [see Levenshtein, 1966], which calculates how many characters you need to remove, insert or replace in a source word to reach the target word. Using Levenshtein and LCSR returned the most accurate trees, correctly stating, for example, that Finnish is not related to any other language and that Italian, Spanish and Portuguese are closely related. However, it also made mistakes such as saying French and English are closely related. Even with mistakes these results are acceptable and easier to achieve than having someone manually checking if two languages are related.

2.2 Other Techniques to Find Translations to Unknown Words

Apart from cognates, there are various other techniques that can be used to translate unknown words. In this section we analyze some possibilities found on various works.

[1] www.europarl.europa.eu

2.2.1 Using Analogy

[Langlais and Patry, 2007] use proportional **analogy** to find translation of unknown words. Proportional analogy can be denoted as $[A : B = C : D]$, which reads 'A is to B as C is to D'. This is adapted to translation systems.

For example to translate the French word *futilité*, they first search for some analogies in French containing that word:

$[activités : activité = futilités : futilité]$

$[hostilités : hostilité = futilités : futilité]$

Secondly, they find English translations for all of the words in those analogies apart from the unknown word:

Activités $< - >$ actions;

Activité $< - >$ action;

Hostilités $< - >$ hostilities;

Hostilité $< - >$ hostility;

Futilités $< - >$ trivialities, gimmicks.

They then create analogies between these, based on the French analogies:

$[actions : action = gimmicks :?]$

$[hostilities : hostility = trivialities :?]$

Reaching that *gimmick* and *triviality* are possible translations of the word *futilité*.

Languages with rich morphology can be dealt with in an analogy way as well, as explained by [Arora et al., 2008]. In that paper, Hindi-to-Japanese translation is investigated using the characteristic that words in Hindi can be looked at as lemmas or as inflectional forms. These inflectional forms can easily be built by simply replacing the characters at the end of the words with other suffixes. For example, the adjective *kAlA* (black) is masculine, it can easily be turned into feminine as *kAlI* and to plural as *kAle*. This is possible because in Hindi, all adjectives end in *A* if it is masculine, just as feminine end in *I* and plural end in *e*. Taking advantage that all the nouns, verbs and adjectives have the same inflections, it makes it easy to construct a set of rules to find similar words.

2.2.2 Using Definitions or Auxiliary Languages

[Eck et al., 2008] explain a way to find definitions for the unknown words on the source side and translate those definitions instead. As an example, the sentence *'**revelan** you have diabetes'* is given, intending to show that the system does not know how to translate the word **revelan**. Looking in a **dictionary** for the definition of the word, they came across *revelan:*

descubrir lo secreto. Knowing how to translate all the words in the definition, the authors choose to show the user a possible translation of the unknown word:

*'(UNK: **revelan** # undiscovered it secret) you have diabetes'*

In this way the user is given a possible translation of that word and can decide on if it fits and what the semantic of the sentence is. One of the main problems encountered was the occurrence of other unknown words within the definitions. A solution proposed is to, instead of choosing the first definition of a list, choose the one that has the least amount of unknown words so that its translation is possible.

[Callison-Burch et al., 2006] also suggest a similar approach by replacing the unknown words with **paraphrases**. To discover paraphrases they use a method suggested by Bannard and Callison-Burch (2005) that uses bilingual parallel corpora to find these paraphrases. For example to find a paraphrase of 'under control', first a phrase with these two words is translated into German, resulting in 'unter kontrolle'. Then it checks for all the phrases in German that contain 'unter kontrolle' and looks at the equivalent English translation to find other translations of 'unter kontrolle', for example 'in check'. These translations are considered paraphrases of 'under control'. This can be very useful in the case where multilingual texts are available. In the case that the translation of 'under control' to a target language is not known, what is done is search for paraphrases in other languages to find one with known translation in the target language. In the example given before, maybe the phrase 'in check' has a known translation from English to the target language and so that translation can also serve as a translation to 'under control'.

In a similar way, [Mann and d. Yarowski, 2001] use bridge languages as a middle way to translate from source to target. **Bridge languages** are the ones that have a lexicon with translations between itself and the source language. When the source word is translated to the bridge language it should make it easier to determine the cognate in the target language.

As an example, when trying to translate the word *bait* in English to the word *isca* in Portuguese, it can be seen that the two words have little in common and so a cognate detection system might not consider them as cognates. To solve this, the authors use Spanish, Italian and other Romance languages as bridge languages. So they translate the source word (*bait*) into these bridge languages, resulting in *esca* in Italian, *carnada* in Spanish and *nadă* in Romanian. They then use cognate detection between these words and Portuguese words. Using this, instead of just cognate detection with the source words, increases the chances of finding a correct cognate, since the Romance languages have more similarities with Portuguese than English. In the example, the Italian word *esca* is closer to the Portuguese word *isca* than the source English word *bait*.

The authors use an adaptation of Levenshtein Distance to determine cognate words between the bridge and target languages. When all these are obtained, each bridge language has a rank of all the most plausible target words to be a translation of the source word. To de-

cide between all of these possible words, the method that achieved the best results chooses the word with the best distance with any of the bridge words and, if there is a tie, choose the word with the lowest combined rank in all of the bridge languages.

2.2.3 Using Context

[Fung and Yee, 1998] show that it is possible to determine which word is more likely to be a translation of another word by comparing the **contexts** of each of the words in two corpora. By finding words with already known translations it is possible to see how many times a certain word appears in both words' contexts and then, if the contexts are similar, it can be assumed that they are a possible translation of each other. The context is assumed to be the the closest words around the selected unknown word. The authors try to achieve translations between English and Chinese and notice that the context for the word *flu* in Chinese is much more similar to the context of the word *flu* in English than to the context of the word *Africa* in English. This shows that context is a good indication of what words are possible translations.

2.2.4 Translating Without Parallel or Comparable Corpora

[Koehn and Knight, 2002] attempts to build a translation lexicon without either parallel or comparable corpora, simply using texts in German and English that have nothing to do with each other. In order to do this they group several different techniques already used in previous papers:

- **Identical words**: words that are exactly the same in both languages such as *immigration* or *computer*. The authors also use rules, similar to the orthographic cues mentioned in [Mulloni and Pekar, 2006], such as replacing *-tät* with *-ty* so that words like *Elektrizität* and *electricity* can be considered the same. Words under the length of 6 are not counted as translation pairs because words of small size have a higher probability of having the same orthography purely by accident;
- **Similar spelling**: if the words are not written exactly the same way, they might still be cognates, like *freund* and *friend*. The authors use LCSR to determine which words should be considered translation pairs;
- **Context**: as used previously by [Fung and Yee, 1998], it is possible to gather a lot of information by the context in which a word is inserted. So with the words obtained from the previous two clues it is possible to define a context for all the other untranslated words both in English and in German and compare the contexts to decide which ones are more likely to be related;
- **Similarity**: in this case the authors want to find relations between words such as *Wednesday* and *Thursday* so that they can find the same relation in the words *Mittwoch*

and *Donnerstag*. This can be achieved in a similar way as the context similarity, but instead between words in the same language;

- **Frequency**: this is based on the principle that the more frequent words in one language should translate into the more frequent words in another language. So the authors do a count for word frequency among those that do not have a translation yet and try to relate them. This is very imprecise and has very poor results.

[Kondrak, 2001] shows another example of an attempt at finding cognates without any sort of corpora, mainly aimed at lesser-studied languages that do not have the abundance of texts that languages such as Portuguese and English do. Even though it focuses on finding cognates, it is a generic technique that ends up finding translations for unknown words even if it does not find its cognates. As a first approach they use the normal phonetic similarities, checking for orthographic clues as to if the words are related or not. On top of that, the author bases the algorithm on vocabulary lists, with a list of words in the two languages and their meanings in English (called glosses) it is possible to determine which words are more likely to be cognates. This is referred to as semantic similarity. In this way English is being used as a sort of bridge language, where both languages' words are being compared by their respective glosses.

In the paper, the author makes use of WordNet [Fellbaum, 1998] to find relations between all of the word glosses such as generalization, metaphor, metonymy, etc. It also has to take into consideration that glosses might differ even when words are cognates, examples given are morphological differences (*ash* and *ashes*), adjectival modifiers (*small stone* and *stone*), synonymy (*grave* and *tomb*) or small semantic changes (*fowl* and *turkey*) among others. The impact of this on the results of the system is not as big as expected.

2.3 Translation of Named Entities

A cognate detection system can be used for other applications such as detecting if two Named Entities (NEs) are the translation of each other. An NE can be the name of a person, location, organization, among other things. NEs are not easy to translate because new ones are introduced frequently and so keeping up to date with all the names and their translation can be a complicated problem.

By running the cognate detection algorithm between two NEs, it can detect if these are a translation of each other or not, simply by seeing if most of the words in the NE in the source language are cognates of words in the NE in the target language.

As an example, giving the following NE:

- *North Atlantic Treaty Organization*;

- *Organização do Tratado do Atlântico Norte*.

The system could validate that they are translations because it would find the following cognate pairs:

-*North/Norte*;

-*Atlantic/Atlântico*;

-*Treaty/Tratado*;

-*Organization/Organização*.

Even if the words are in a different order, the fact that most of the words are cognates is an indication that these two NEs are possible translations. [Carvalho, 2010] developed a system similar to the one described here to find NE translations.

Parallel and comparable corpora can also be used to find the translations between NEs. Parallel corpora is much easier to use, because its texts are written as side-by-side translations, so it is possible to find the translation of a certain NE from English to Portuguese simply by finding the NE in a text in English and looking at the same line in the parallel Portuguese text. However parallel corpora is hard to find and thus comparable corpora is a possible replacement. While parallel corpora is a direct translation, comparable corpora is a group of texts in different languages that talk about the same topic. An example can be found in news websites where the news are covered in different languages. [Hassan et al., 2007] make use of both these types of corpora to extract named entity pairs out of Arabic and English texts.

The algorithm for comparable corpora starts by aligning the documents. All the Arabic texts are translated into English so that the keywords of these texts may be chosen along with the keywords of the English texts. They then use WordNet to find semantic similarities in these keywords and distribute the words in different clusters. Finally they count how many times words in a certain cluster appear in the Arabic text and in the English text and use a formula on entropy to calculate if these two texts can be considered comparable corpora.

After doing all this they start looking for the NE translation pairs in these comparable texts by using Editex [see Zobel and Dart, 1996], phrase tables or a combination of both.

Editex is an edit distance, but instead of treating all the characters as different, it takes advantage of their phonetic similarity (for example *b* and *p* are phonetically similar, while being different from *l* or *r*). This way, a replacement of the character *b* by *p* is treated as a smaller edit distance than a replacement of the character *b* by *l*.

Using parallel corpora the algorithm is much easier: it simply goes through all the names in Arabic and tries to find a translation in the English text and then does the same from English to Arabic, attributing different weights depending on if a translation pair appears in both translations or just in one of them.

Even though it is a complicated task to maintain, [Huang and Vogel, 2002] attempt to build an NE dictionary in an iterative manner. They first use a baseline model to extract all

the possible NE translations and their respective probabilities in texts in both English and Chinese. From this list, they choose the pair (NE Chi, NE Eng) with the best probability and then remove all other associations with that NE Chi and NE Eng, repeating this process until there are no more NE pairs. After running the baseline model, there are still some tagging problems, so the authors run a corrected algorithm to retag the original corpora. With the new tags they rechoose the best probability for each with the new algorithm, constructing a new dictionary each time. They stop the iterative process when the new dictionary does not bring clear improvements on the previous one.

The same Chinese symbol can be found in different names of people from different countries, for example, Jin (Chinese), Kim (Korean) or King (English) are all represented by the same Chinese symbol. [Huang, 2005] clusters different languages that have similarities in order to be able to correctly translate these names. The clustering of languages is done with a list of bilingual name translation pairs where the language origin of each name is known. Then, the distance between languages is calculated based on the distance of these pairs. The most closely related languages are then grouped together. This allows the grouping of languages such as Spanish and Portuguese due to their similarities and the separation of languages such as Korean and English that have very different names.

Another way to translate unknown NEs is through web searches of the context. [Al-Onaizan and Knight, 2002] use Google [2] to find candidates for translation of NE from Arabic to English. The system first translates all the known words in the Arabic corpus and introduces some of the key words in the search engine. It then searches in the documents returned for NEs that can be possible translations of the unknown NEs. The choice of which translation to choose can be done by counting how many times each candidate appears in the results and choosing the top scorer.

[Jiang et al., 2007] use the same technique of using web search to achieve the desired translation of NE, but with a slight addition to improve English/Chinese translation. It divides the English NE into syllables according to linguistic rules defined by the authors. It then tries to find a Chinese character to each of these syllables to have an overview on how the target NE would look like. When using the web search it then constructs the query with the English NE and one of the Chinese characters that are a translation of the English NE's syllables. This is done once for each of the Chinese characters obtained and then the results are combined together to make the statistical analysis.

[Ling, 2010] suggests that instead of using full web documents, which require a long time to process and can be very large, the *anchor texts* can be used alone to achieve similar results. The *anchor text* is the text that is visible on a clickable link. The author claims that most of the times these *anchor texts* contain a good summary of the information contained in the web document opened with that hyperlink, especially with NEs. A link relating to a

[2] www.google.com

page of a person probably has the name of that person in the clickable link, same happens with other Named Entities such as movies or book names.

The concept behind working with *anchor texts* to achieve NE translation revolves around the fact that two *anchor texts* that correspond to the same link, even if in different languages, tend to contain the same information and therefore the same NEs.

The algorithm works in 4 steps:

1) Given an NE in the source language, it is used as a query to retrieve a list of documents, in the target language, that contain it;

2) From each of those documents, it looks for all the *anchor texts* in it;

3) Generation of candidate target NEs by extracting all n-grams in these anchor links;

4) The algorithm scores each candidate and returns the most probable translation.

The scoring is made with the help of various features, with frequency being the most important one. If a candidate appears a large number of times, the likelihood of it being the correct translation is higher; but since there is a number of link texts that are very common in web documents such as “click here” or “in english”, the author has added a global frequency feature to counter balance the previously mentioned frequency feature. The global frequency refers to how many times that candidate appears in all of the *anchor texts*. So the chosen candidate should be the NE that maximizes the frequency while minimizing the global frequency. Other features are word distance (difference between the number of words in the source NE and the candidate), NE recognition features (gives a value to the probability of the candidate being an NE or not), language detection feature (checks if the candidate is in the target language), co-occurrences feature (counts the number of times the source NE appears in the same link as the target NE) and link occurrences feature (counts the number of times a candidate is the entire *anchor text*).

2.4 Summary

In the various papers introduced, different methods to finding translations of words were introduced. Most of the methods to determine if two words are cognates or not make use of coefficients to find the distance between two words. Examples of these coefficients include the Dice coefficient [see Dice, 1945], LCSR or the Levenshtein Distance [see Levenshtein, 1966]. An aid to these distances is the use of rules, such as the ones introduced by [Mulloni and Pekar, 2006].

Other ways to translate words that do not include calculating the distance between two words were also introduced:

- Translating the definition or a paraphrase of the unknown word [see Eck et al., 2008];

- Using auxiliary languages (often called bridge languages), so that you can translate from the source language to the auxiliary language and then from this one to the target language [see Callison-Burch et al., 2006, Mann and d. Yarowski, 2001];
- Comparing the context of words to see which ones are more closely related [see Fung and Yee, 1998];
- Using analogy to compare the unknown words to words with known translation [see Arora et al., 2008, Langlais and Patry, 2007].

A system that correctly determines pairs of words that are translation of each other can then be used to other ends, such as finding how closely related two languages are of each other [see Rama and Borin, 2011] or translating NEs [see Al-Onaizan and Knight, 2002, Ling, 2010].

3 Baseline System

As previously stated, the system that we developed was based on a previous work done by Luís Carvalho [Carvalho, 2010].

In his thesis, he has built a cognate detection system, where, given two words, the system will tell if those words are cognates of each other or not, based on a set of similarity measures and with the help of transliteration rules. If two words are considered cognates of each other it is assumed that they are a good translation for each other.

3.1 Baseline Architecture

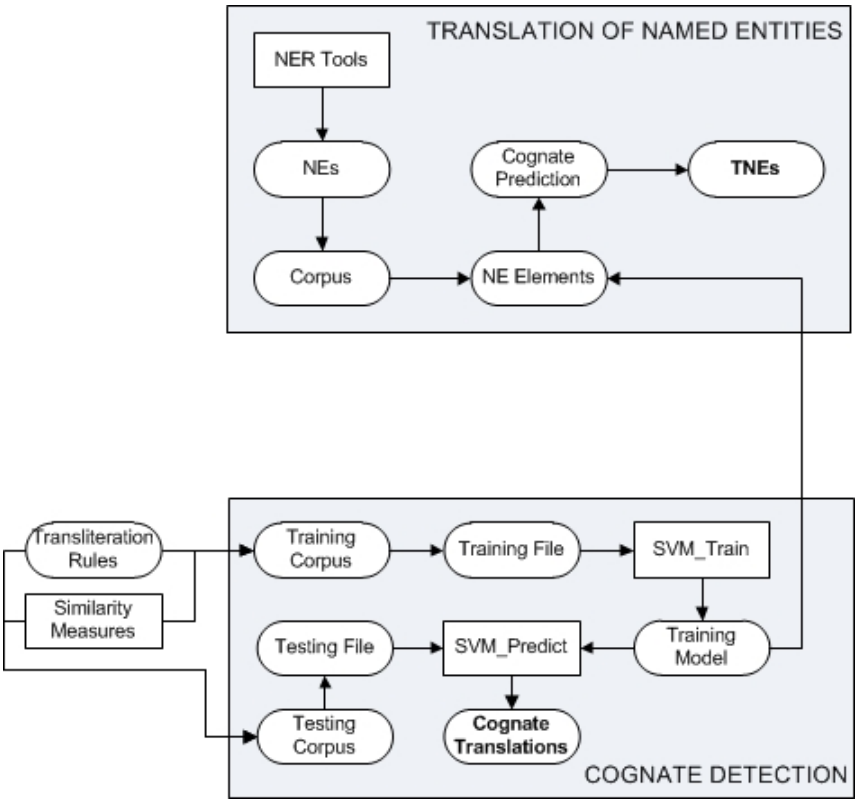

Figure 1: Baseline System Architecture.

The architecture for the baseline framework is shown in Figure 1. The system is based on comparing words in corpora through a set of 10 different similarity measures. These measures are explained in detail in Appendix A. Each word is first parsed by a group of transliteration rules as explained in Section 3.2.

After comparing the words, the system will output ranked pairs of words that will then be used in a Support Vector Machine (SVM) to generate a model which is used in the training phase. Using this model, the system can then predict if a different pair of words are cognates

or not.

To translate NEs, the system first uses an NE recognition tool on the corpus. Then with the resulting NEs and with the training model obtained in the cognate phase, it predicts which NEs are translations of each other by counting how many cognate words each pair has. This is further detailed in Section 3.3.

3.2 Transliteration Rules

Along with similarity measures, the system uses transliteration rules in a similar way that [Mulloni and Pekar, 2006] uses orthographic cues. These rules determine the substitution of certain prefixes, suffixes or substrings in the middle of words by a single character [char], in the case that some patterns in a source language frequently translate into a certain pattern in the target language. The objective of this substitution is to make certain pairs of words have a better result on the similarity measures, by saying that some of their patterns are the same. As an example, the suffix *tion* in English normally translates into the suffix *ção* in Portuguese (*addition, adição* or *petition, petição*); so the suffix would make the scores of the word pairs be lower when in fact it is common to see those suffixes as translations of each other. This way those words would be seen by the system as *addi[char]* and *adi[char]* resulting in better scores and a bigger probability of the words to be considered cognates.

This can also occur with prefixes such as *hyper* in English translating normally to *hiper* in Portuguese (*hyperactive*, *hiperativo* or *hypersonic*, *hipersnico*) or even in patterns found in the middle of the words like replacing *mm* with *m* (*command*, *comando* or *immigration*, *imigração*).

However, when replacing patterns in the middle of the word, it should be taken into consideration the neighbouring letters of these patterns, in the example of *command* and *comando*, in both cases the letters to the left and right of the replaced pattern are *o* and *a*. On the other hand, if we think of the words *command* and *comendo*, the words are not cognates, so it would not make sense to apply the same rule of replacing *mm* with *m*. By looking at the pattern's neighbouring letters, we can see that in the word *command*, the pattern's neighbours are *o* and *a*, while in the word *comendo*, it has neighbours *o* and *e*. Since they do not have the same neighbours, the rule should not be applied.

After calculating these scores, the system uses SVM and the library *libsvm* [3] to build the training model. In this case, the SVM was trained with scores of the similarity measures for a pair and the result of if those words are cognates or not. Then, in the testing phase, when faced with a new pair of words, it can calculate the measures for this pair and predict if the corresponding words are cognates or not.

[3] http://www.csie.ntu.edu.tw/ cjlin/libsvm/

3.3 Translating Named Entities

This framework is also used to translate Named Entities (NE). To find the NEs in a text, it uses the Stanford NER [4] to recognize all the English entities; to recognize the Portuguese entities two recognizers were tested, XIP NER and an adaptive method created by the author. This adaptive method runs as explained in Algorithm 1.

Algorithm 1 Adaptive Method to Recognize NEs

$SetNEs \leftarrow \emptyset$
$NE \leftarrow \emptyset$
$currFunctionWords \leftarrow \emptyset$
for $word \leftarrow nextWord() != EOF$ **do**
  if $word$ starts with capital letter **then**
    $NE \leftarrow NE \cup currFunctionWords \cup word$
    $currFunctionWords \leftarrow \emptyset$
  else if $NE \neq \emptyset \wedge word \in fileOfFunctionWords$ **then**
    $currFunctionWords \leftarrow currFunctionWords \cup word$
  else
    $SetNEs \leftarrow SetNEs \cup NE$
    $NE \leftarrow \emptyset$
    $currFunctionWords \leftarrow \emptyset$
  end if
end for
if $NE \neq \emptyset$ **then**
  $SetNEs \leftarrow SetNEs \cup NE$
end if

As an example, if the system catches the word *United*, it will set $NE = \{United\}$ and keep on reading. When it comes across the word *States*, since it is a word that starts with a capital letter, it sets $NE = \{United\ States\}$. The next word would be *of*, it does not start with a capital letter, however, since it is a function word (the list of all function words is defined in an external file) and there is an NE being built, it will be added to the set of current function words, so $function = \{of\}$. By reading the next word as *America*, the system will add both this word and the function word to the NE, $NE = \{United\ States\ of\ America\}$. Finally, if it reads *has* as the next word, considering this neither starts with capital letter nor is a function word, it discards the word and adds the NE *United States of America* to the set of all NEs found.

Then, the NE translator uses the cognate detection algorithm to compare pairs of NEs in English and Portuguese. It retrieves all of the cognates in the texts to be compared and then if one NE is mostly composed by cognates of another NE, those two are considered a translation of each other.

[4] http://www-nlp.stanford.edu/ner/

For example, with *United States of America* and *Estados Unidos da América*, since the system will say (*United*, *Unidos*), (*States*, *Estados*) and (*America*, *América*) are cognates of each other, it will consider these two NEs as translations.

3.4 Summary

Our created system started with a baseline system explained in [Carvalho, 2010].

This work used cognate detection to determine if two words were translations of each other or not.

To achieve this, the author uses transliteration rules and similarity measures to calculate the similarity of words, then with the aid of a Support Vector Machine it determines which words are possible translations.

This system is then used to translate Named Entities (NEs), by determining the number of cognates between two NEs in different languages.

4 Towards Unknown Words Translation

This chapter describes some changes to the baseline system in order to improve the translation of unknown words. The changes made in terms of cognate detection are explained in Section 4.1. In addition to the cognate module, we have implemented two new modules, one that focuses on translation by analogy and another one using context, detailed in Sections 4.2 and 4.3 respectively. To reach the final decision on which is the most likely translation of an unknown word, we take into account the results of all these modules. Section 4.4 talks about how to merge the results in order to rank all the possible translations.

4.1 Cognate Detection Module

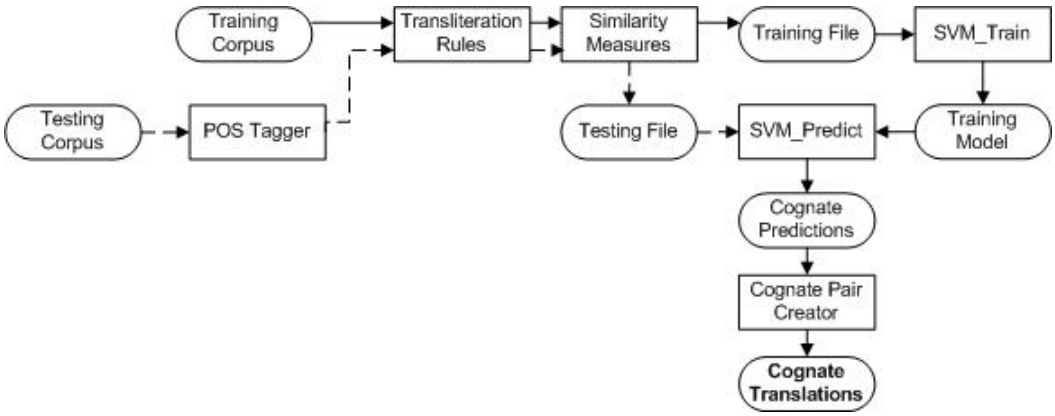

Figure 2: Proposed Cognate Detection System Architecture.

Our proposed change to the system is shown in Figure 2.

The main change occurs before the testing file can go to the prediction phase. Instead of comparing each of the words in the testing file to check if they are cognates, the files go through a POS Tagger named TreeTagger [5]. The tagger separates each words in groups according to their lexical category (Name, Verb, Adjective, etc.), when creating the word pairs with the similarity measures, only words with the same category are compared. This guarantees that even if a name and a verb are similar, they are not considered cognates of each other.

Another change implemented, in order to speed up the process of figuring out translations of a word, was only to compare two words if their first three characters are the same. This drastically cuts down the time the system takes to execute.

Both these changes lower the number of false cognates found, although also lowering the number of correctly guessed cognates. However, we will see in the Evaluation section that, with the help of the other modules, this decrease in correctly guessed cognates is not

[5] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/

crucial for the final results.

4.2 Analogy Module

The system we describe here targets the translation of unknown words via the analogy paradigm. Consider that A is an unknown word. If we have detected an analogy of the form [A : B = C : D] – which reads as "*A is to B as C is to D*" – and assuming that we know how to translate B, C and D, we can find a possible translation of A (to achieve this, we assume that a bilingual lexicon is given as input). For instance, having found the unknown word *instruct*, it is possible to create an analogy as the one represented above with known words, such as [instruct : instruction = construct : construction]; furthermore, knowing all the translations of the known words, we can create a similar analogy in Portuguese [? : instrução = construir : construção]; from this we can infer that *instruir* is a possible translation for the word *instruct*.

In an off-line process, the system learns a set of rules, which represent the prefixes or suffixes that can be used in order to establish analogy relations. In the following we describe how this set of rules is learned from a monolingual corpus and how it can be used to find analogies between words.

4.2.1 Learning analogy rules

The first thing the system does is to go through all the word pairs in the monolingual corpus and find which rules can be extracted from each pair.

4.2.1.1 The rule's syntax

Rules are written in the form $[remove]\backslash[insert]$, where remove and insert represent the characters that need to be removed and inserted into a word to transform it into the other word. For example, when comparing the word pair (*stays*, *staying*), the rule to transform "stays" into "staying" is "s\ing", which means removing the character 's' and inserting "ing".

In the case there are no characters to remove or insert, the corresponding side of the rule is replaced by '$'. For instance, in the word pair (*stay*, *stays*), the only change that needs to occur is the insertion of the character 's', so the rule would simply be "$\s".

The way we chose to differentiate between the representation of suffix rules and that of prefix rules is to have the rule to the right or to the left of a '|'. The '|' represents the part of the word that does not change. If the characters displayed in the rule were removed from the end of the words (thus being a suffix from these words), the rule is to the right of '|'. In the two cases presented above the rules would then be "|s\ing" and "|$\s", respectively. If, on the other hand, the characters were removed from the beginning (a prefix of the words), the rule would be presented to the left of the '|'. One example is the pair (*difference*, *indifference*) where the rule would be "$\in|". This is used to help differentiate from suffix and prefix rules

further on in the system.

4.2.1.2 Finding the longest common subsequence between two words

These rules are built by creating a table that is used to calculate what the longest common subsequence (lcs) between two words is. To do this, a table is initialized with both the first column and the first line being composed of only zeros as shown in Table 1.

		S	T	A	Y	I	N	G
	0	0	0	0	0	0	0	0
S	0							
T	0							
A	0							
Y	0							

Table 1: Lcs table between the words “Stay” and “Staying”, before the algorithm is applied.

Then, it follows the dynamic programming Algorithm 2 that helps in calculating the number of equal consecutive characters between the two words up to a certain point.

Algorithm 2 Sequence table algorithm

for $i := 1 \rightarrow w1.length$ **do**
  for $j := 1 \rightarrow w2.length$ **do**
    if $w1\,[i] = w2\,[j]$ **then**
      $seq\,[i]\,[j] := seq\,[i-1]\,[j-1] + 1$
    else
      $seq\,[i]\,[j] := 0$
    end if
  end for
end for

By following the algorithm using the words in Table 1 we would obtain Table 2.

		S	T	A	Y	I	N	G
	0	0	0	0	0	0	0	0
S	0	1	0	0	0	0	0	0
T	0	0	2	0	0	0	0	0
A	0	0	0	3	0	0	0	0
Y	0	0	0	0	4	0	0	0

Table 2: Lcs table between “Stay” and “Staying”, after the algorithm is applied.

In this case we can see that the lcs is 4 characters long, composed of the characters “stay”.

4.2.1.3 Extracting Rules

If the two words’ **lcs is less than 3 characters long**, they are not considered to be similar enough and therefore no rule is made between the two words. Also, to prevent two long words from being similar just because they have an lcs of 3 characters or more, **the remaining characters that are not part of the lcs can be a maximum of 6**. The way we concluded that these would be the optimal values can be read in Section 5.2. For example, if we consider the words “lone” and “longitudinally”, the lcs between them is 3 characters long: “lon” and there is only one remaining character in the word “lone”, however, “longitudinally” has a total of 11 characters that are not part of the lcs and breaks the second condition of comparison, thus the system does not try to obtain a rule between the two words.

If the two words pass the tests mentioned above, they are candidates to analogies. In that case, using the generated table, the system will try to figure out a rule that can turn one word into the other.

The first thing to do is to check if the rule associated with words $w1$ and $w2$ is a prefix or a suffix. To do this we follow Algorithm 3. With this algorithm we can see that simply by checking two cells of the table obtained with Algorithm 2 we can determine what type of rule it is.

Algorithm 3 Algorithm to determine if a rule is a prefix or a suffix.

$l1 := w1.length$
$l2 := w2.length$
$seq :=$ Algorithm 2 between words w1 and w2
$lcs := seq[0][0]$
if $seq\,[l1]\,[l2] = lcs$ **then**
  $rule := prefix$
else
  if $seq\,[lcs]\,[lcs] = lcs$ **then**
    $rule := suffix$
  end if
end if

Suffix rules

We are now going to use the example of the words “canto” and “cantar”, where the rule is to remove the ‘o’ from the end of the word and insert the suffix “ar”.

w1 = “canto”

w2 = “cantar”

lcs = “cant” (length: 4)

Following Algorithm 3, we check the last cell of the table to see if the value is the same as the lcs.

		C	A	N	T	A	R
	0	0	0	0	0	0	0
C	0	1	0	0	0	0	0
A	0	0	2	0	0	1	0
N	0	0	0	3	0	0	0
T	0	0	0	0	4	0	0
O	0	0	0	0	0	0	0

Table 3: Lcs table between “Canto” and “Cantar”, prefix test.

As shown by Table 3, the last cell has the number 0 which tells us this rule is not a prefix. So we check the cell [lcs.length][lcs.length] to verify if it is a suffix.

		C	A	N	T	A	R
	0	0	0	0	0	0	0
C	0	1	0	0	0	0	0
A	0	0	2	0	0	1	0
N	0	0	0	3	0	0	0
T	0	0	0	0	4	0	0
O	0	0	0	0	0	0	0

Table 4: Lcs table between “Canto” and “Cantar”, suffix test.

Looking at Table 4, we can confirm that this cell contains the number 4, which is equivalent to the number of characters in the lcs. This means that the first four characters of each word are the lcs and thus the rule should be a suffix rule.

This being the case, we mark the characters to remove as the remaining characters from w1 and the characters to insert as the remaining characters from w2, as shown in Table 5.

Reaching the rule “|o\ar”.

Prefix rules

If we look at the example of the words “real” and “unreal”:

w1 = “real”

w2 = “unreal”

lcs = “real” (length: 4)

		C	A	N	T	A	R
	0	0	0	0	0	0	0
C	0	1	0	0	0	0	0
A	0	0	2	0	0	1	0
N	0	0	0	3	0	0	0
T	0	0	0	0	4	0	0
O	0	0	0	0	0	0	0

Table 5: Sequence table between “Canto” and “Cantar”, characters to remove and insert.

Following the same algorithm and looking at the last cell of the table, we get Table 6.

		U	N	R	E	A	L
	0	0	0	0	0	0	0
R	0	0	0	1	0	0	0
E	0	0	0	0	2	0	0
A	0	0	0	0	0	3	0
L	0	0	0	0	0	0	4

Table 6: Sequence table between “Real” and “Unreal”, prefix test.

The number in this cell is the same as the length of the lcs, therefore this rule is a prefix. In this case we skip that same number of cells diagonally as shown in Table 7.

		U	N	R	E	A	L
	0	0	0	0	0	0	0
R	0	0	0	1	0	0	0
E	0	0	0	0	2	0	0
A	0	0	0	0	0	3	0
L	0	0	0	0	0	0	4

Table 7: Sequence table between “Real” and “Unreal”, skipping cells.

The rule is then constructed as before, getting the remaining characters of both words. Since there are no remaining characters in w1 (as seen in Table 8), the rule obtained is “$\un|”.

4.2.1.4 Storing “similar” words and scoring rules

Each of the rules created previously is then stored by the system, being associated to a list of all the word pairs that have generated it. Using the example above of the words “real” and “unreal”, apart from the rule “$\un|”, “un\$|” is also inserted to count for the case where w1

		U	N	R	E	A	L
	0	0	0	0	0	0	0
R	0	0	0	1	0	0	0
E	0	0	0	0	2	0	0
A	0	0	0	0	0	3	0
L	0	0	0	0	0	0	4

Table 8: Sequence table between “Real” and “Unreal”, characters to remove and insert.

is “unreal” and w2 is “real”.

After checking every word pair in the corpus, a file is created with all the rules that have occurred in more than 2 word pairs and with the respective word pairs printed next.

The next step is to store these rules and their respective word pairs in such a way that it is made easy to check what rules can be applied to a new word. One option would have been to use Suffix Trees [see Ukkonen, 1995, Weiner, 1973] (one example of its use with bilingual lexicons is given by [J. Costa and Russo, 2011]). However, this method focuses on creating one suffix tree per word and in our case we want only two trees: one to contain all the information we need of possible suffixes and another to contain all the information of possible prefixes.

The method we ended up choosing for storage is very similar to a Trie, as explained in [Fredkin, 1960]. A Trie is a tree structure that allows strings with the same prefix to have a similar path. In Figure 3 we show an example of this, where the words “do”, “did”, “big” and “dig” are introduced in the same Trie. With this we can see that if two words start with the same sequence of characters, less nodes are needed to represent those two words in the Trie. This can be very helpful in our case since we need to store a large number of rules, many of which are associated with the same prefix or suffix to be removed.

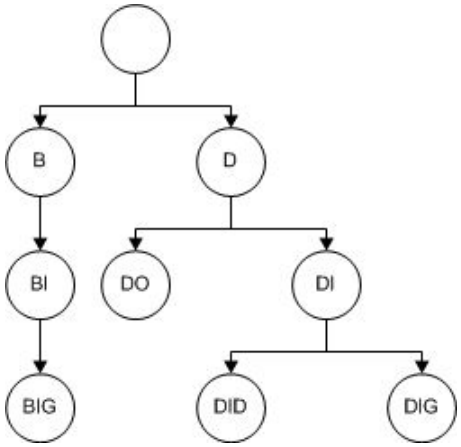

Figure 3: Example of a Trie with the words “do”, “did”, “big” and “dig”.

Using the previously created file that lists all the rules, we can start building the prefix and suffix Tries. Instead of inserting words, like in a normal Trie usage, we insert the sequence of characters that we want to remove. This way, each node represents a prefix or a suffix that can be removed from a certain word as stated by the equivalent rule.

In the beginning we have only two nodes: the origin of the prefix Trie and the origin of the suffix Trie. Each level in the Trie corresponds to the number of characters in the remove part of the rule, starting with 0 in the origin nodes. These nodes will include all the rules where no characters are removed, where only insertions occur.

Each node in a graph is represented by a string of characters that are meant to be removed if this node is reached (denoted _sub). For example, if we encounter the rule "|o\ar", we create a new node in the first level of the tree that is represented by 'o' and pointed by the suffix origin node (Figure 4).

The nodes also keep:

- The character that is needed to transit from the previous node's rule to this one (_trans);
- A list of strings to insert after removing the characters (_ins);
- The previous node (_prev);
- A list of the next nodes (_next);
- A hashtable that associates each string to insert with a list of word pairs that follow the corresponding rule (_wordPairs).

Using the rule "|o\ar" as an example, we only have two nodes. Their details are listed in Table 9.

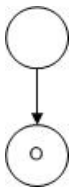

Figure 4: Trie after insertion of node O.

The reverse rule also needs to be inserted, that means "|ar\o" also needs to be inserted in the tree. Since this is a suffix rule, we need to start by creating the node corresponding to the last letter of the string to be removed, which means node R, and then keep adding letters behind that until we have the full string to be removed. If it was a prefix rule we would have started by using the first letter of the string to be removed to create the node and would keep moving forward from there. In this case we only need to insert nodes R and AR (Figure 5). These two nodes and the changes they enforce in the already existing nodes can be seen in Table 10.

Node	OriginS	O
_trans	‘’	‘o’
_sub	“”	“o”
_ins	$\emptyset$	$\{\text{“}ar\text{”}\}$
_prev	*null*	OriginS
_next	$\{O\}$	$\emptyset$
_wordPairs	$\emptyset$	$\{\text{“}ar\text{”} :< canto, cantar >\}$

Table 9: List of nodes after insertion of node O.

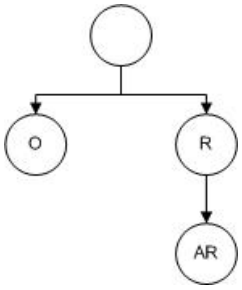

Figure 5: Trie after insertion of nodes R and AR.

Furthermore if we also had the words “viver” and “vivo”, the system would find the rule “|er\o”. The nodes that would need to be created would be R, ER and O, but since both R and O already exist, the only node that needs to be added to the Trie is ER (Figure 6). The addition of the node ER and the changes to the nodes R and O are represented in Table 11.

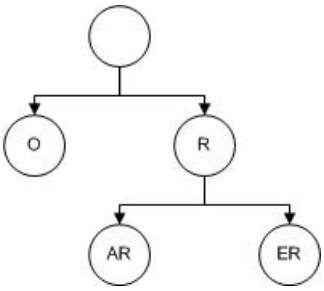

Figure 6: Trie after insertion of node ER.

4.2.2 Translating unknown words via analogy

When we find an unknown word, the system goes down both the prefix and the suffix Trie, finding all the rules that can be applied on this word.

For instance, let us consider that “*saltar*” had been selected as an unknown word and we had the suffix tree represented by Table 10.

Starting in the node *OriginS*, there are no rules associated with it, so it is not possible to

Node	OriginS	O	R	AR
_trans	‘’	‘o’	‘r’	‘a’
_sub	“”	“o”	“r”	“ar”
_ins	$\emptyset$	$\{\text{“}ar\text{”}\}$	$\emptyset$	$\{\text{“}o\text{”}\}$
_prev	*null*	OriginS	OriginS	R
_next	$\{O, R\}$	$\emptyset$	$\{AR\}$	$\emptyset$
_wordPairs	$\emptyset$	$\{\text{“}ar\text{”} :< canto, cantar >\}$	$\emptyset$	$\{\text{“}o\text{”} :< cantar, canto >\}$

Table 10: List of nodes after insertion of nodes R and AR.

Node	O	R	ER
_trans	‘o’	‘r’	‘e’
_sub	“o”	“r”	“er”
_ins	$\{\text{“}ar\text{”}, \text{“}er\text{”}\}$	$\emptyset$	$\{\text{“}o\text{”}\}$
_prev	OriginS	OriginS	R
_next	$\emptyset$	$\{AR, ER\}$	$\emptyset$
_wordPairs	$\{\text{“}ar\text{”} :< canto, cantar >\}$ $\{\text{“}er\text{”} :< vivo, viver >\}$	$\emptyset$	$\{\text{“}o\text{”} :< viver, vivo >\}$

Table 11: Nodes with changes after insertion of node ER.

make any analogies at this point. Then, the system looks at the last character of this word, which is ‘r’, so it searches through the nodes that are in the _next list of *OriginS* to find one that has the character ‘r’ as the _trans, finding node *R*.

We move to that node and proceed in the same way. Again there are no rules associated with this node. We move on to the previous letter of the word, getting the character ‘a’. By searching node *R*’s list of next nodes we find node *AR* that has the character ‘a’ as the transition character.

When reaching node *AR*, there is one string in _ins. This means that there is one rule that can be applied after removing the characters “ar”, in this case, it is inserting the character ‘o’. By applying this rule to the word “*saltar*” we reach the word “*salto*”. The system then checks if the word “*salto*” exists on the lexicon and thus has a known translation so that it can aid in the discovery of the translation of the word “*saltar*”. After finding out that the word “*salto*” is known, we can start building an analogy: [saltar : salto = ? : ?].

In the case where the word resulting from applying a suffix rule does not exist in the lexicon and if the last character of the word is a vowel, we search in the same lexicon for any word that is similar to the one created, with the single difference being that the last character is a different vowel. This is a valid change since sometimes when a rule is meant to remove a certain vowel from the end of a word, the same rule can actually be applied to any word ending in a vowel. For instance, consider we have found the word “*excelentíssimo*” as an

unknown word and the only rule that fits it is "|íssimo\o", from the word pair (*muitíssimo*, *muito*). The resulting word from applying that rule is "*excelento*", which is not a valid word. However, if instead of looking for the word "*excelento*" in the lexicon, we look for "*excelent*" followed by a single vowel, we find the word "*excelente*". With this information we can try to continue the analogy process with the analogy [excelentíssimo : excelente = muitíssimo : muito].

To replace the question marks on the analogy, we use the word pairs that are associated to the replacement we just made. In order to do that, the system consults the hashtable _wordPairs of that node. By looking for the list of word pairs associated with the characters that were inserted in the word, we get the word pair (*cantar*, *canto*), getting the final analogy of [saltar : salto = cantar : canto].

The next step is to find translations for each of the known words in this analogy. This means finding translations for "*salto*", "*cantar*" and "*canto*". By looking at the same lexicon, we find that "*salto*" can be translated as "*jump*", while "*cantar*" can be translated as "*singing*" and "*canto*" can be translated as either "*sing*" or "*corner*". With this we can build two separate analogies: [? : jump = singing : sing] and [? : jump = singing : corner].

Then we repeat the starting process, which compares two words and tries to find a relation between them. First, it creates the lcs table for both (*sing*, *singing*) and (*corner*, *singing*). Since (*corner*, *singing*) share only the character 'n', their respective lcs is only 1 and thus the analogy containing that word pair is immediately discarded. The word pair (*sing*, *singing*) passes this test and so we try to find out what rule is used to go from one word to the other. Using the same method that was explained previously, we obtain the rule "|$\ing".

We then apply this rule to the only known word on the left side of the analogy, which is the word "*jump*". Finally, we obtain a possible translation for the original unknown word ("*saltar*") as the word "*jumping*".

If we have a much larger number of rules and therefore a bigger Trie than the one presented here, we end up with a big range of words that we can compare to our unknown word. In that case, instead of comparing the word "*saltar*" with just the word "*salto*", we could have maybe compared it with other words, such as "*saltando*" or "*saltos*", while also obtaining words that did not make sense and were not in the original lexicon such as "*saltars*" (by applying the common rule of "|$\s"). However, these words would be discarded for not having a translation.

We would also have a larger number of word pairs associated to each rule which would create a bigger number of analogies. In that case, there might be more than one possible translation to the unknown word. To determine a score of each of these translations we use four different factors which attribute scores to each analogy (analogies have the form [A : B = C : D], translated to [A' : B' = C' : D']):

- The number of characters that were replaced in word A to reach word B;

- The number of characters that were replaced in word B' to reach word A';
- The number of equal characters between words B' and D', starting from the end if the rule applied is a suffix rule or from the beginning if it is a prefix rule;
- If word A' exists in the lexicon, the score of this factor is 2, otherwise it is 1.

By adding factors 1 through 3 and multiplying this sum by factor 4, each of the analogies has its own score. Summing all of the scores of the analogies that have the same word A', we can attribute a score to that word and, with that, compare it with other possible translations to decide which one is the most adequate.

4.3 Context Module

The other completely new module we have implemented is the one that is based on context. It works similarly as to what is explained in [Fung and Yee, 1998]. The general idea surrounding the use of context is that, if two words are related, there is a good probability that the words before and after each word are the same too. We can use this knowledge to find translations of words in parallel or comparable corpora.

As an example, consider the word "*fisherman*". Not knowing the meaning of this word, we can look at its context. The word is normally inserted in sentences such as "*the fisherman caught the fish*" or "*the fish brought in by a fisherman*". By using these sentences we can create a context list for the word "*fisherman*":

- "*the*" - 3 times
- "*caught*" - 1 time
- "*fish*" - 2 times
- "*brought*" - 1 time
- "*in*" - 1 time
- "*by*" - 1 time
- "*a*" - 1 time

If we look at the respective comparable corpus in Portuguese, we would most likely find the sentences "*o pescador apanhou o peixe*" and "*o peixe trazido por um pescador*". If we created the same context list for each of the words, we would find that the word "*pescador*" has the following:

- "*o*" - 3 times
- "*apanhou*" - 1 time

- “*peixe*” - 2 times
- “*trazido*” - 1 time
- “*por*” - 1 time
- “*um*” - 1 time

Using a bilingual lexicon, just as the one used for the analogy phase, we can see that a lot of these words' translations appear in the context list of the word “*fisherman*”. By stating this, we can also state that it is very likely that the words “*fisherman*” and “*pescador*” are possible translations of each other, given the similar contexts that they are inserted in.

Now the question is how many words to catch. If we caught the whole sentence the word is inserted in, that would give us a higher chance of getting all the key words that relate to it. However, that would not only create a large context list, but also would catch many words that are also not related to the one we are finding the context of. The goal is to maintain a certain balance between the size of the list and the number of words caught to the left and to the right of the unknown word. After some preliminary experiments, we opt to catch only the 2 closest words on each side, as the closer words would be the ones most relevant to the unknown word, also guaranteeing the context list would be of a reasonable size.

By looking at the previous example though, we see that this restriction misses some of the keywords. For example, in the sentence “*the fish brought in by a fisherman*”, the only words that would be added to the context list of the word “*fisherman*” would be “*by*” and “*a*”, which tell us nothing about the word itself. In the end, the context list of “*fisherman*” would look like this:

- “*the*” - 2 times
- “*caught*” - 1 time
- “*by*” - 1 time
- “*a*” - 1 time

Within these words, only the word “*caught*” would actually be relevant to include in the context. Looking at the other words we can also see that they are mostly uninteresting for context related purposes, as these words appear in numerous sentences and can be used in almost all contexts. Therefore, a way to improve the context list without increasing its size is by ignoring some of the most common words. To do this we use the Tree Tagger again. Before calculating the context of each word, we tag them with their lexical category. Any word that is not an adjective, an adverb, a name or a verb is then skipped and the 2 words on each side that do belong to one of those classes are added to the context list.

Looking back at our example with the word “*fisherman*”, if we take into consideration this new idea to create the context list, we would have the following:

- “*caught*” - 1 time
- “*fish*” - 2 times
- “*brought*” - 1 time

This list is much more relevant to distinguish the word “*fisherman*” from other words.

After creating the context list for each of the words, the system normalizes all of these lists. This means that instead of having the number of times each of the words appear in that context, it shows the percentage of times that word appears (calculated by dividing the number of times a word appears, by the total number of times every word appears in the context), when compared to all of the words in that list. Using the previous example we then have:

- “*caught*” - 25%
- “*fish*” - 50%
- “*brought*” - 25%

We do this to prevent that words that appear many times in the corpus are considered translation of all other words. For example, the word “*is*” appears in numerous sentences and, therefore, has a much bigger number of entries in the context list than a word such as “*fisherman*”. If the context list of *is* had one entry of each word: “*caught*”, “*fish*” and “*brought*”, it could be considered similar to the word “*pescador*”. However, if instead we normalize each entry, each word has a much lower score and is less likely to be associated to “*pescador*”. This way we guarantee that a words’ high frequency will not turn into this same word being considered a valid translation numerous times.

After normalizing each list, the next step is to look for the unknown words among all of these. To do this we use the bilingual lexicon; this lexicon is needed for two different parts of this module: one of them is to find unknown words, and the other is to compare two contexts in different languages to see how similar they are.

Each of the unknown words will have its context list compared to the context lists of all words in the opposite language. By looking at each word in each list and its respective percentage, we can calculate the similarity between two context lists.

Using the context list for “*fisherman*”, shown above, and the context list for “*pescador*” as follows:

- “*apanhou*” - 25%
- “*peixe*” - 50%
- “*trazido*” - 25%

With the lexicon, we can see that "*apanhou*" is the translation of "*caught*", "*peixe*" is the translation of "*fish*" and "*trazido*" is the translation of "*brought*". So, by looking at the two context lists, we see they are exactly the same and thus these two words are likely to be a translation of each other.

4.4 Merging All Modules

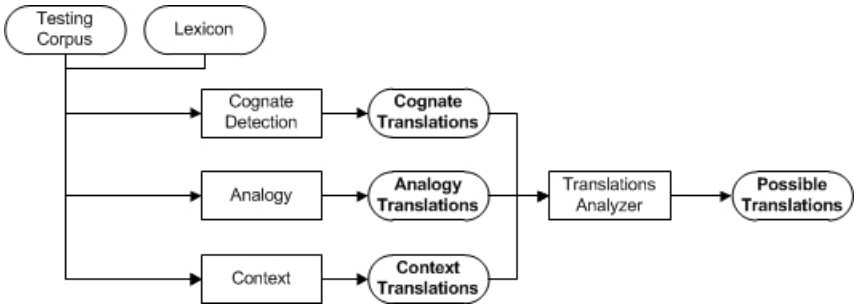

Figure 7: Tentative Architecture of the whole System.

Each of the modules has its own proposed translations and equivalent scores. To decide what the most likely translations of a word are, we implemented a system that merges the scores of each module. This merging can be seen in Figure 7.

4.4.1 Scores Calculation

The final decision on what is the best translation to a given unknown word will be taken by the Translations Analyzer. This module will see the output of all 3 methods used in the system and, taking into consideration each result, output a ranked list of the most probable translations for the unknown word.

After the system's evaluation (described in depth in Section 5), we came to the conclusion that the context module does not output viable results. This fact, along with the added time it takes for this module to calculate a score, led us to decide to cut the context module from the final system.

Since the cognate evaluation is based on words that exist in the lexicon, many of the valid analogy words would not have a cognate score associated to them which would end up lowering their final score. For instance, the word *"assistants"* is a valid translation of the word *"assistentes"* and, as we will see in Section 5.2.4, the analogy module does suggest it as a translation. However, since the word *"assistants"* does not exist in the lexicon, the corresponding score of the cognate module is 0. Due to this, we have decided to run the cognate model not only over all the words in the lexicon, but also over all the words suggested by the analogy module. Since the words *"assistants"* and *"assistentes"* are written in a similar way, the cognate module will attribute a high score to this pair, which will end up raising its

overall score. The final architecture of the system is shown in Figure 8.

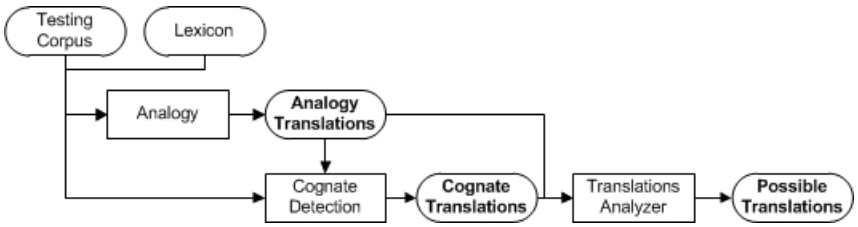

Figure 8: Final Architecture of the whole System.

4.4.2 Finding Translations Online

To create an interactive web service that executes the system, we made use of the functionalities of Apache Tapestry 5 [6]. Using this framework, we created a website, shown in Figure 9.

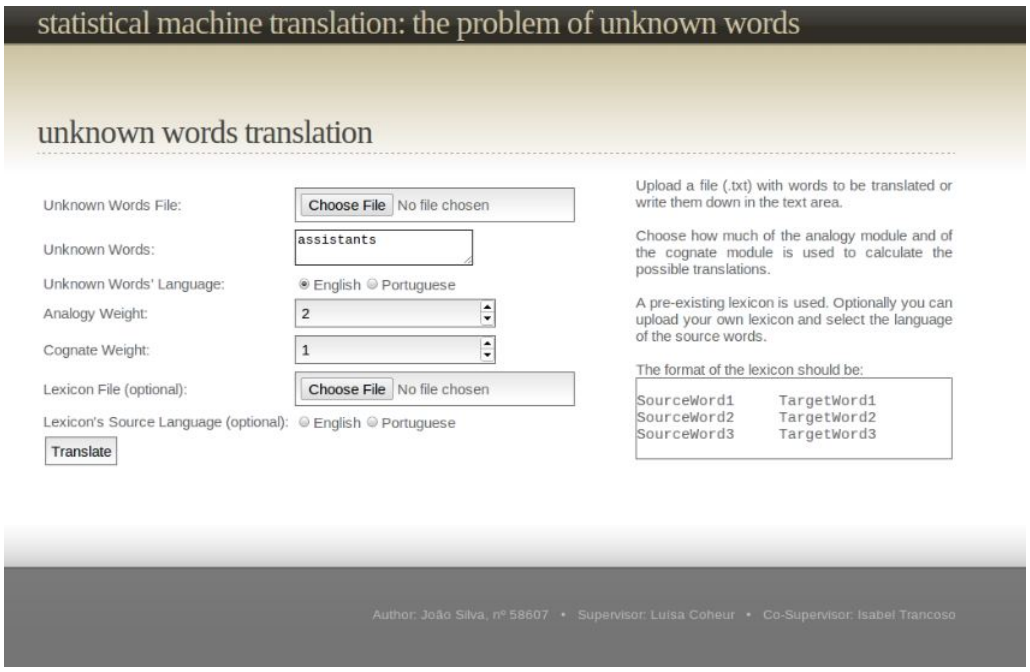

Figure 9: Index page of the online system.

In this website, we allow users to upload a file containing words to be translated or write these down in a text area. The system has an existing lexicon with 20000 entries, but the user might choose to upload his own lexicon, specifying if it has Portuguese or English words in the first column.

Additionally, users can choose the weights of the analogy module and of the cognate module, being able to decide if the translation is determined entirely by one of these modules

[6] http://tapestry.apache.org/

or by a combination of the two.

Depending on the size of the lexicon and the number of unknown words to translate, the system might take up to a minute to show the results. This time also depends on whether the user chooses to use the cognate module or not, since this module takes several times longer to complete than the analogy module.

An example of the screen showing the results can be seen in Figure 10.

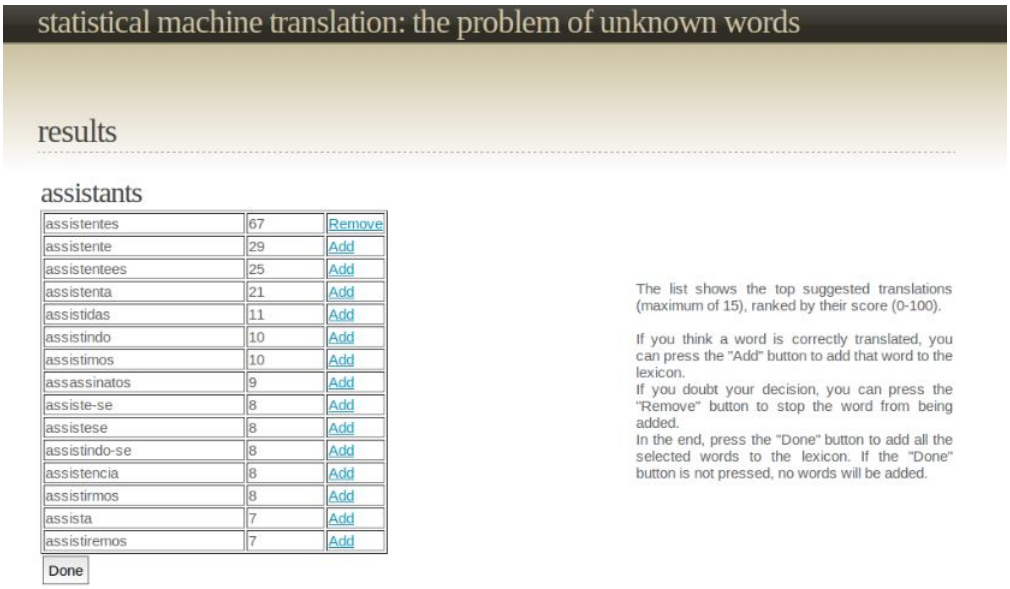

Figure 10: Results page of the online system.

In this screen, the top 15 translations of each unknown word are shown, ranked according to an attributed score that ranges from 0 to 100. If the user decides that a suggested translation is correct, he can add this pair (*unknown Word*, *suggested Translation*) to the lexicon, by clicking the “Add” button. This button will then turn into a “Remove” button; in case the user thinks he has made a mistake in considering that translation as correct, he can click that button and remove the pair from the lexicon. Once the user’s decisions are final, he can click the “Done” button and the user will be back to the index page.

4.4.3 Finding Translations Offline

Alternatively, the system can be executed from the command line, having the same options as mentioned in the previous section. The command should be written as follows:

```
java merger {unknownWordsFile} [-pt|-en]
        {analogyWeight} {cognateWeight} ({lexiconFile} [-pt|-en])
```

Where “unknownWordsFile” is the file that contains the words that are to be translated, followed by the language of those words (either “-pt” for Portuguese or “-en” for English);

“analogyWeight” and “cognateWeight” are a non-negative number that represents each module’s weight on the results; and “lexiconFile” is the file with the lexicon to be used, followed by the language of its first column. Just as in the online version, the lexicon is optional and, if no lexicon is provided, the default lexicon will be used.

It is also possible to add words to the lexicon through the command line using the following command:

```
java merger -a {word1} {word2} [-pt|-en] ({lexiconFile} [-pt|-en])
```

This will add the pair (*word1*, *word2*) to the lexicon given in “lexiconFile”; again, if this entry is omitted, the default lexicon will be used. The first language entry in the command refers to the language of word1.

The offline method gives the user more freedom to select any two words to add, instead of relying on words that were considered suggestions of an unknown word. Such freedom can be costly though, for instance a spelling mistake such as:

```
java merger -a assistants assistantes -en
```

Will introduce the incorrect pair of words in the lexicon, which might not be immediately visible by the user. If the mistake is visible, it is possible to remove the pair from the lexicon, using a similar command:

```
java merger -r {word1} {word2} [-pt|-en] ({lexiconFile} [-pt|-en])
```

If the pair (*word1*, *word2*) exists in the lexicon, it will be removed.

4.5 Summary

The created system consists of three modules:

- **Cognate Detection**: We introduced a POS Tagger to the baseline system, forcing it to only compare words that have the same lexical category. Along with that, two words are only considered cognates if their first three letters are the same.
- **Analogy**: This module uses logical analogy to find the translation of words. Having an unknown word A, we can create an analogy of the form [A : B = C : D] (which reads as “*A is to B as C is to D*”) and if we know how to translate B, C and D, we can find a possible translation of A.
- **Context**: The third module of the system checks the words surrounding the unknown word and compares those words to other words’ surrounding words. If two words are related, and therefore possible translations of each other, then there is a good probability that the surrounding words are also related.

In the end, we tried to merge the results from these three modules, in order to maximize the number of words correctly translated. However, the context module did not produce good enough results and we decided to only include the first two modules, with the analogy module counting for 75% of the results and the cognate module for 25%.

5 Evaluation

In this section we show various evaluations we have made to the system. To do so, we start by showing individual evaluations of each model; by doing this, we can find the strengths and weaknesses of each one separately. Finally, we do an evaluation of the system as a whole to see what results it offers when all the modules are taken into account.

5.1 Cognate Detection Module

Evaluation of the cognate detection, on the baseline system referred in Section 3, is made with 19 economic news and 19 politic news extracted from the Euronews website [7], both in English and in Portuguese. 15 of each are used for training and the other 4 are used for testing. For NE translation, all 19 are used for testing because, since it is based on the cognate detection algorithm, no training phase is needed. The choice of which news to use and the annotations of the valid cognates was made by [Carvalho, 2010] during his thesis work. We are going to use the same news for the evaluation of this module, so that we can directly compare our results with those achieved in the baseline.

5.1.1 Using the POS Tagger

One of the changes introduced to the system is the addition of the POS Tagger. In the case that two words have different lexical classes, they are not even considered as cognates and, therefore, will not be compared; this is checked both before the training and the testing phase.

Even though it seems simple to compare two lexical classes, the complexity of the problem comes from the fact that the Portuguese tagger outputs different tags when compared to the English tagger. The Portuguese tagger is much simpler than the English one, having, for example, only one tag for all verbs, while the English tagger has seven different classes of verbs. To make both taggers work with one another, all the English tags had to be mapped to only one Portuguese tag. The resulting associations can be seen in Appendix B. The only English tag that had to be mapped to two different Portuguese tags was the *Predeterminer*. In the document containing all tags and their explanation, some examples of words in this class are *all*, *both* and *many*. For example, the word *all* can be translated as *totalidade*, which is considered by the Portuguese tagger as a *Noun*, or translated as *todos*, which is considered an *Adjective*. Therefore, we decided to consider these words to be both a *Noun* and an *Adjective*.

The number of word pairs that were compared, before and after the POS Tagger was added to the system, can be seen in Table 12. The number of word pairs checked decreases

[7] http://www.euronews.net/

to only around 30% of the number of comparisons made on the baseline system, which means that many pairs that are most likely not cognates of each other are discarded earlier.

Economics Corpora				
Corpus	# News	# Word pairs before	# Word pairs after	Percentage
training	15	107809	31899	29.59%
testing	4	35688	10446	29.27%
Politics Corpora				
Corpus	# News	# Word pairs before	# Word pairs after	Percentage
training	15	156453	48213	30.82%
testing	4	41300	11642	28.19%

Table 12: Comparison between the number of word pairs checked before and after the introduction of the POS Tagger.

This decreases the number of false positives found, which are the pairs that were considered cognates when in fact they are not. This is shown in Table 13, by the increase in the precision of the system. In this table, we can also see that the number of cognates missed has increased, shown by a slight decrease in recall. This is due to the fact that the tagger can make mistakes when attributing a tag to a word in Portuguese, ending up with a different tag than the one it attributes to the corresponding word in English. Errors like this can later be corrected in the Analogy and Context phase. Combining these scores and calculating the *F-Measure*, we can see an overall improvement of the system.

	Precision	Recall	F-Measure
Before Tags	71.7%	55.9%	62.8%
After Tags	76.5%	55.36%	64.3%

Table 13: Comparison between the statistics of cognates before and after the introduction of the POS Tagger.

5.1.2 Using Cognates for Named Entity Translation

Another of the evaluations made consisted in running the already existing NE Translation model through a set of 2 years of news from Euronews, both in English and in Portuguese, to try to collect a good number of pairs of NEs that are possible translations of each other.

After that, we went through a text of one Portuguese Broadcast New and annotated all of the NEs that could be found in the text. In total there were 119 NEs, between names, locations and other types of NEs. Then, we attempted to find out how many of these 119 NEs could be found in the pairs obtained from the Euronews corpora. This allowed us to see how the system behaved in a real situation.

The results were nowhere close to the ones that were expected. Only 31 out of those 119 NEs (26%) could be found in the Euronews corpora, even though many of those that were not found were Portuguese names that should not be translated. Some of the weirder cases that did not appear in the Euronews corpora were countries such as “Estados Unidos” or “Espanha”.

Out of those 31, 18 of them (58%) were the same in English and Portuguese, which means that leaving them untranslated would have the same result. In other 4 (13%) the only difference was the accents in the Portuguese language. That leaves us with only 9 out of the original 119 (7.5%) that had changes in their translation:

Protecção Civil -> Protection Civile

União Europeia -> Europe

Roma -> Rome

John Kennedy -> Kennedy OR John F Kennedy

Japão -> Wen Jiabao

Partido Socialista -> Socialist Party

Hélio Loureiro -> Helio Lourieo

Manchester United -> Manchester OR United

Gaza -> Gaza Strip

Out of these, “Protecção Civil”, “União Europeia” and “Manchester United” have a wrong translation even though it does not stray too far from the truth, but “Japão” has a completely unrelated translation. “John Kennedy”, “Hélio Loureiro” and “Gaza” could have been left the same in the English equivalent and therefore only 2 out of the original 119 (1.6%) are usable.

With these results we can see that the training needs to be done with a large number of corpora, because even with 8109 pairs of English and Portuguese news, the results obtained were very weak. There is a lot that can be done here to improve the results. For example, the training could also be more specific, depending on if we want locations, people names or other types of named entities. This ensures that more NE translations are learned during the training phase to increase the percentage of correct translations made.

5.2 Analogy Module

The analogy module can be considered the key addition to this system. Through a method using rules, it tries to find the most likely translations to an unknown word. Since there are various steps to this process, the evaluation of such system needs to be divided into different areas, in order to guarantee that all are working accordingly or to find where the most common mistakes are made.

In order to fulfill this step-by-step evaluation, we divide it into 3 categories:

- Section 5.2.2 evaluates if the words that are considered to be word pairs are in fact related or if their graphic similarity is just a coincidence;
- Section 5.2.3 looks into the rules that are created and discarded;
- Section 5.2.4 shows how varying the weight of some scores can improve the results and gives a final evaluation of the analogy module.

5.2.1 Evaluation Setup

The evaluation of this module was made using Europarl parallel texts. To create the bilingual lexicon, we have used a Portuguese corpus containing 10000 lines from an Europarl text and extracted all the unique words from it. To have a legitimate translation of all these words, we then inserted them all into Google Translate [8] and created a bilingual lexicon from it. The same was made with the parallel English corpus containing the same 10000 Europarl lines. Finally, we merged both lexicons obtained, by joining all the translations in those two lexicons, and removed all the duplicates to achieve one single bilingual lexicon with almost 20000 entries, that could be used to test the analogy module.

For the testing phase we also use Europarl. This time we extracted 100 new lines in both English and Portuguese. We then tried to match each of the words in these corpus to an entry in the bilingual lexicon. If the word existed, it was a known word and, therefore, not interesting to us; however, if the word did not have an entry in the lexicon, this meant it was an unknown word and we could try to find a translation to it through the analogy module. We obtained a total of 44 Portuguese and 9 English unknown words. From those, we started by using 10 Portuguese and 4 English as a development set, so that we could study and test them. Our goal with this development set was to tune the factors and their respective scores in order to achieve the best results possible.

5.2.2 Word Pair Evaluation

As explained before, there are two conditions to determine which word pairs are considered by the system as similar words and which are not. For two words to be considered a word pair, they need to have a minimum number of lcs characters ($min\|lcs\|$) and a maximum number of non-lcs characters ($max\|\overline{lcs}\|$). While sorting through word pairs, the system should only build rules with those where the two words are actually related, while discarding all those pairs where the words' meanings have nothing in common.

In this phase of the evaluation, we simply used the lexicon to extract all the word pairs that followed the two requisites, varying the value of $min\|lcs\|$ between 2 and 4 and the value of $max\|\overline{lcs}\|$ between 3 and 6.

[8] http://translate.google.com

5.2.2.1 Results From Portuguese Word Pairs

First, we ran this evaluation for Portuguese word pairs. The total number of word pairs can be seen in Table 14, as well as the percentage of these word pairs where the change between the words occurs in the suffix, for instance, in the word pair (*stay*, *staying*) (the percentage of word pairs where the change occurs in the prefix is the remaining percentage).

$min\|\|lcs\|\|$	$max\|\|\overline{lcs}\|\|$	Total Number	Suffix Changes (%)
2	3	97610	48.2%
2	4	302712	36.9%
2	5	769570	33.3%
2	6	1656288	32.5%
3	3	56840	54.8%
3	4	143055	42.5%
3	5	323043	36.0%
3	6	624371	33.0%
4	3	39120	63.4%
4	4	82362	50.3%
4	5	154942	39.4%
4	6	262809	31.8%

Table 14: Number of Portuguese word pairs with varying prerequisites.

As shown, the number of word pairs grows as we make these requirements less strict. However, what is also possible to see is that the number of word pairs with suffix changes increases (and therefore the number of word pairs with prefix changes decreases) as these requirements get stricter. A reason for this is that prefix word pairs are a lot less accurate than suffix word pairs. For instance if we look at the word pairs (*restam*, *gastam*) and (*restos*, *gastos*), the system can assume that, in both cases, the rule "re\ga|" can be applied to transform one word into the other, when in fact the words in these word pairs are not related.

It could be said that if the system also had the word $gastar$ and the unknown word that was found was the word $restar$, the analogies [restar : gastar = restam : gastam] and [restar : gastar = restos : gastos] would be valid. However, this is not what we are looking for, as with that group of words it should be sufficient to have the analogies [restar : restam = gastar : gastam] and [restar : restos = gastar : gastos], which can all be made by other suffix rules. Therefore we would like the system to find the rules "|ar\am" and "|ar\os" instead of the rule "re\ga|". This is how we are going to count valid and invalid pairs.

From the resulting list of word pairs we randomly chose 100 of them, for each combination of $min||lcs||$ and $max||\overline{lcs}||$, and manually checked if the words are in fact related or if, although having similarities, they can not be considered word pairs in the sense that we need them to be. The results can be found in Table 15.

These results confirm that the prefix word pairs are very rarely correct. In fact, only 3 out

$min\|lcs\|$	$max\|\overline{lcs}\|$	Prefix	Suffix
2	3	0.0%	58.8%
2	4	0.0%	56.1%
2	5	0.0%	26.7%
2	6	0.0%	10.8%
3	3	2.9%	69.7%
3	4	0.0%	69.4%
3	5	1.5%	33.3%
3	6	0.0%	34.4%
4	3	0.0%	97.0%
4	4	1.9%	85.1%
4	5	0.0%	71.4%
4	6	0.0%	58.8%

Table 15: Percentage of Portuguese word pairs that are valid.

of the 695 prefix word pairs manually checked were actually related words. An example of a word pair that was considered valid is (*satisfeito*, *insatisfeito*), where clearly one word is made by adding a prefix to the other word.

On the other hand, the suffix word pairs achieve good results in most cases, getting less and less accurate as the number of word pairs increase. We can see the poor results achieved by setting the $min\|lcs\|$ at 2, reaching almost as low as 10% valid suffixes in the case where $max\|\overline{lcs}\|$ is set at 6. However, even though it might be tempting to just assume we should just set the variables at 4 and 3 respectively, because it achieves the best results, we must consider also that those values do not even give us 40000 word pairs, which means that, even though we are discarding many invalid word pairs, we might also be discarding others that could be crucial for the analogies later on. In the analogy phase, the more word pairs we have to compare the words with, the better.

5.2.2.2 Results From English Word Pairs

Similarly to what we have done for Portuguese word pairs, we extracted all English word pairs from the lexicon, obtaining the values in Table 16.

It is possible to see that the number of word pairs, in general, is higher than in Portuguese and also that the percentage of suffix word pairs follows a similar pattern, even though all the percentages are slightly lower. This denotes an overall increase in the percentage of prefix word pairs.

A similar decrease in the percentages can be seen in Table 17, where we show the results of manually checking the 100 word pairs extracted for each combination of the variables.

The results in English are considerably lower than the ones obtained for Portuguese word

$min\|\|lcs\|\|$	$max\|\|\overline{lcs}\|\|$	Total Number	Suffix Changes (%)
2	3	237343	44.3%
2	4	668625	34.5%
2	5	1494748	30.1%
2	6	2833246	27.0%
3	3	109263	43.3%
3	4	269416	31.0%
3	5	559431	24.5%
3	6	994057	20.7%
4	3	57192	50.2%
4	4	103717	40.1%
4	5	173407	33.0%
4	6	272446	27.1%

Table 16: Number of English word pairs with varying prerequisites.

pairs, maybe due to the fact that the English words are somewhat smaller than Portuguese ones and thus more words will be considered similar under the conditions we tested them. However, there is an increase in prefix word pairs, having a total of 16 valid ones out of the over 750 found.

Overall, we can conclude that having the variable $min\|lcs\|$ set at 2 achieves poor results, despite giving us the biggest range of word pairs to choose from and, therefore, a bigger dataset to create analogies. However, from this analysis we can not firmly point which values should be given to the variables, since even with many wrong results caught by the system, many of these might be discarded later in the rule making process for not having another word pair with the same rule as that one. For example, the word pair (*evening*, *covening*) was considered similar by the system, being applied the rule "ev\cov|". This rule does not seem to be valid, however, since no other word pair will have that output as a rule, the word pair (*evening*, *covening*) will never actually be used by the system, which means that even if it is wrong and the system right now thinks it is a valid word pair, it will later discard it. So in order to reach a better decision on what values to choose, we need to evaluate the rules created by gathering all these word pairs.

5.2.3 Rule Evaluation

Results from the word pair evaluation have shown us that the stricter the variables $min\|lcs\|$ and $max\|\overline{lcs}\|$ are, the more accurate results we achieve. However, we also eliminate a large number of word pairs while making these results more accurate, leading to a possible loss of other word pairs that would be considered valid. To have a better understanding of the results, we need to group these word pairs into rules, which is the next step of the system.

$min\|lcs\|$	$max\|\overline{lcs}\|$	Prefix	Suffix
2	3	4.8%	32.4%
2	4	0.0%	13.5%
2	5	0.0%	2.9%
2	6	0.0%	5.4%
3	3	8.0%	42.0%
3	4	0.0%	31.0%
3	5	0.0%	28.6%
3	6	0.0%	13.3%
4	3	10.0%	76.0%
4	4	3.8%	63.8%
4	5	2.9%	61.3%
4	6	0.0%	54.5%

Table 17: Percentage of English word pairs that are valid.

Each word pair has a rule associated to it, but each rule might have more than one word pair, which is what we evaluated next. We have extracted all the rules from each word pair and counted how many times each rule appears. We have separated each rule into three categories:

- Rules with only one word pair associated to them;
- Rules with two word pairs associated to them;
- Rules with three or more word pairs associated to them.

We started by deciding that all rules with only one word pair associated would be discarded, since many of these words in the word pairs are, most likely, not related. We also wanted to separate the rules that have two word pairs, since these are the ones that can be encountered in the border and we wanted to be sure most of them had valid results, therefore, making it worth it to count these as valid, or if we could also discard this set of rules and save some time in the system.

Just like in the word pair evaluation, we extracted 100 rules from each category, in Portuguese and English, and manually checked which rules were valid and which could be discarded.

Results are shown in the next sections.

5.2.3.1 Results From Portuguese Rules

In Table 18 we can see the number of rules that each of these categories has and the percentage of suffix rules in each of them. It can be seen that many word pairs were immediately

		1		2		>2	
$min\|\|lcs\|\|$	$max\|\|\overline{lcs}\|\|$	Total #	Suffix (%)	Total #	Suffix (%)	Total #	Suffix (%)
2	3	107676	38.0%	13420	29.2%	22876	31.0%
2	4	434636	34.0%	40948	19.4%	64848	21.1%
2	5	1228644	34.4%	97130	13.3%	147274	13.8%
2	6	2780702	35.1%	191082	9.7%	284862	9.5%
3	3	47028	34.8%	5814	24.7%	11492	30.0%
3	4	170320	33.1%	18992	18.0%	33004	22.8%
3	5	458388	34.1%	45232	13.2%	73572	15.6%
3	6	963018	34.3%	87732	9.2%	136500	10.5%
4	3	23672	33.1%	3136	23.0%	6766	33.7%
4	4	80132	30.9%	9824	20.0%	18094	29.5%
4	5	192906	28.8%	21512	15.0%	35756	21.7%
4	6	372920	25.9%	38584	10.8%	59630	15.6%

Table 18: Percentage of Portuguese rules that are valid with one, two or more than two word pairs associated with them.

discarded, due to the nonexistence of any other word pair with the same rule associated to it. In average, from the rules discarded for only having one word pair associated with it, a third were suffix rules.

We were expecting a much larger number of prefix rules to be discarded due to the poor results achieved from prefix word pairs, but if we look at the example given before of the word pairs (*restam*, *gastam*) and (*restos*, *gastos*), we can see that if these four words exist, the rule "re\ga|" has two word pairs associated with it.

On the other hand, looking at the number of rules with two word pairs, the number of prefix rules is bigger, presumably for the reason explained above. In this case, the percentages work in a similar way that they did in the word pair evaluation, increasing the number of prefix rules as $max||\overline{lcs}||$ increases, but not having a drastic change when $min||lcs||$ changes.

When we analyze the number of rules with more than two word pairs, we see that there is an increase in the number of rules when compared to the ones with only 2 word pairs associated. This is a good indication that most of the rules the system accepts should be valid, due to the high number of word pairs. It is also possible to see, when comparing the two columns, that the percentage of suffix rules also increases slightly if we have more than two word pairs associated to it.

For the rules with 2 word pairs, all rules have the same weight in the system, but in rules with more than 2, it does not make sense to count them all as one unit, as each rule should have a different weight depending on the number of word pairs associated with it. For example, the rule "|ty\ly" has 5 word pairs associated with it (for instance (*safety*, *safely*)), while the rule "|ism\$" has 16 word pairs ((*conformism*, *conform*) being one of

them). Previously, we were counting both of these rules as one unit, when in fact, the number of word pairs in each rule should determine the weight it has in the results. So the first rule would have an added value of 5 and the second 16.

We decided then to redo the count of rules with more than two word pairs, this time taking into account the number of word pairs each rule has. This can show us more accurately how many suffix word pairs are in this category, being able to make a better comparison with the suffix percentage of rules with only 2 word pairs. The results can be seen in Table 19.

$min\|\|lcs\|\|$	$max\|\|\overline{lcs}\|\|$	Suffix (%)
2	3	54.5%
2	4	38.0%
2	5	24.1%
2	6	15.5%
3	3	63.2%
3	4	48.8%
3	5	33.5%
3	6	22.7%
4	3	71.8%
4	4	61.8%
4	5	48.7%
4	6	37.7%

Table 19: Percentage of Portuguese suffix word pairs in rules with more than 2 word pairs.

It is clear that that there is a high number of suffix word pairs among these, even surpassing the number of prefix word pairs in some cases. This increase in percentage, when compared with all rules in this category having the same weight, also shows that the suffix rules have in average more word pairs associated to them, meaning that, when making analogies, these will end up having a bigger impact on the results than the prefix rules.

We went through 100 rules in each category to count how many of these were valid rules. We consider rules to be valid when most of the word pairs that are associated to them are also valid, which means that the two words that created that rule have to be related to each other. Results can be found in Table 20.

With these values, we can once more see the poor results obtained from prefix word pairs: only 1 of the rules with 2 word pairs was valid out of the 981 checked and only 2 out of the 892 checked for rules with more than 2 word pairs.

Looking at the suffix columns, we see the very low number of rules that could be considered valid when having only one word pair associated to it, which leads us to conclude that these rules are rightfully discarded. Mainly when running the tests with $min\|lcs\|$ set as two, the results show that most rules that are discarded are actually not useful for the system. As we set $min\|lcs\|$ as three, we see an increase of rules wrongly discarded, especially when

		1		2		>2	
$min\|\|lcs\|\|$	$max\|\|\overline{lcs}\|\|$	Prefix (%)	Suffix (%)	Prefix (%)	Suffix (%)	Prefix (%)	Suffix (%)
2	3	0.0%	5.7%	0.0%	29.2%	0.0%	51.9%
2	4	0.0%	0.0%	0.0%	19.4%	0.0%	62.1%
2	5	0.0%	0.0%	0.0%	13.3%	0.0%	75.0%
2	6	0.0%	0.0%	0.0%	9.7%	0.0%	60.0%
3	3	0.0%	29.2%	1.3%	24.7%	1.6%	84.2%
3	4	0.0%	37.0%	0.0%	18.0%	0.0%	88.5%
3	5	0.0%	6.7%	0.0%	13.2%	0.0%	92.3%
3	6	0.0%	6.8%	0.0%	9.2%	0.0%	61.5%
4	3	0.0%	30.3%	0.0%	23.0%	0.0%	88.9%
4	4	0.0%	18.5%	0.0%	20.0%	1.5%	88.6%
4	5	0.0%	23.1%	0.0%	15.0%	0.0%	92.1%
4	6	0.0%	20.8%	0.0%	10.8%	0.0%	96.9%

Table 20: Results of Portuguese rules with one, two or more than two word pairs associated with them.

the $max\|\|\overline{lcs}\|\|$ is low; having $min\|\|lcs\|\|$ as four achieves consistently average results.

For rules that have two word pairs associated with them, the best results are also achieved when $min\|\|lcs\|\|$ is set at two, not having much lower results in both other cases. However, as we increase the value of $max\|\|\overline{lcs}\|\|$, the number of valid suffixes decreases.

This could lead us to think that perhaps $min\|\|lcs\|\|$ = 2 would be something to take into consideration, but, along with the poor results in the word pair evaluation, we see that in the rules that have the biggest weight in the system, and that therefore will affect the analogies the most, $min\|\|lcs\|\|$ = 2 clearly achieves the worst results, maintaining an average of only two third of suffix rules being valid rules.

When comparing with the much more impressive results of setting $min\|\|lcs\|\|$ as 3 or 4, we see that setting it as 2, despite the good results with rules with 1 and 2 word pairs, is really not the most valid option overall. In general, having the minimum lcs as 4 achieves the best results out of all options, so one more time, being stricter gives us better results. However, we can not be sure that in order to achieve these higher results in the suffix rules, we did not discard other rules that could be valid and thus we need more general results to decide between these two values.

Contrary to the word pair evaluation though, when analyzing the changes the results suffer from changing the $max\|\|\overline{lcs}\|\|$, we realize that, instead of seeing an improvement in the results when the value is set at a lower value, it is the other way around, having higher results the less strict this variable is. The best results of the system are then achieved from $min\|\|lcs\|\|$ = 4 and $max\|\|\overline{lcs}\|\|$ = 6, reaching almost 100% valid suffix rules out of the rules evaluated.

5.2.3.2 Results From English Rules

We ran the same tests for English rules as we did for Portuguese ones. Starting with the number of rules fitting each of the three categories and how many among those rules are suffix rules. The results are shown in Table 21. This time, however, we already take into consideration the weight of each rule giving the number of word pairs in it. This means that in the column of rules with more than 2 word pairs, the percentage is equivalent to the total number of suffix word pairs out of all word pairs in that category of rules.

		1		2		>2	
$min\|\|lcs\|\|$	$max\|\|\overline{lcs}\|\|$	Total #	Suffix (%)	Total #	Suffix (%)	Total #	Suffix (%)
2	3	117210	39.3%	13396	31.8%	20932	45.7%
2	4	475314	30.7%	43664	17.6%	57452	30.0%
2	5	1224798	27.5%	102190	10.7%	125436	18.7%
2	6	2444092	25.8%	205028	6.7%	241678	11.7%
3	3	50992	30.7%	5802	24.5%	9870	49.9%
3	4	195580	22.0%	15604	16.3%	22346	38.7%
3	5	491884	18.3%	31916	11.7%	42074	28.3%
3	6	972830	16.2%	56084	8.4%	71168	20.1%
4	3	22510	25.1%	2580	20.6%	4986	58.8%
4	4	63912	22.8%	6114	16.4%	9882	50.8%
4	5	135908	20.7%	11378	13.4%	16774	41.5%
4	6	248900	18.4%	21570	8.5%	28953	30.5%

Table 21: Number of English rules with one, two or more than two word pairs associated with them.

In terms of overall numbers there is no pattern that can clearly be visible when comparing this table with the equivalent one for Portuguese rules (Table 18). Looking at the percentages though, we can see that they are much less consistent when comparing rules with only one word pair, having a noticed decrease as the $max\|\|\overline{lcs}\|\|$ increases. The values of the percentage of suffix rules, when there are three or more word pairs per rule, are also lower than the same values for Portuguese rules, again showing an increase in prefixes when in English that could already be verified in the word pair evaluation.

Picking 100 rules from each category and checking for their validity, we reached the results shown in Table 22. In this table it is evident once again that suffix word pairs or suffix rules achieve better results than prefix ones, having a total of 14 out of 2788 prefix word pairs checked, which is an improvement from the 3 out of 2701 checked in the Portuguese rule evaluation.

The main characteristic that is visible when comparing this table with Table 20 is that the results achieved in the Portuguese evaluation, both for rules with two or more than two word pairs, were slightly higher for Portuguese than for English suffix rules. In the Portuguese

		1		2		>2	
$min\|\|lcs\|\|$	$max\|\|\overline{lcs}\|\|$	Prefix (%)	Suffix (%)	Prefix (%)	Suffix (%)	Prefix (%)	Suffix (%)
2	3	0%	2.8%	0.0%	7.4%	0.0%	21.6%
2	4	0%	0.0%	0.0%	0.0%	1.4%	37.9%
2	5	0%	0.0%	0.0%	10.0%	1.1%	10.0%
2	6	0%	2.8%	0.0%	28.6%	0.0%	36.8%
3	3	2.7%	11.1%	1.2%	5.3%	1.5%	54.5%
3	4	1.3%	10.0%	0%	7.1%	3.2%	39.5%
3	5	0%	6.7%	0.0%	0.0%	0.0%	60.0%
3	6	0%	0.0%	1.1%	0.0%	0.0%	33.3%
4	3	0%	16.7%	1.2%	33.3%	0.0%	83.3%
4	4	0%	27.8%	0.0%	50.0%	1.5%	71.9%
4	5	0%	9.1%	1.3%	52.0%	0.0%	77.8%
4	6	1.3%	8.7%	0.0%	12.5%	0.0%	85.8%

Table 22: Results of English rules with one, two or more than two word pairs associated with them.

case we saw five cases in which the percentage of valid suffix rules, for rules with more than 2 word pairs, was over 88 %, while here we see that the higher score achieved is 85.8%. This also follows the results seen in the word pair evaluation.

Looking at both tables we reach the same conclusions: first $min\|lcs\|$ set as 2 achieves the worst results in all cases, whether it is in English or Portuguese and also in both evaluation types done so far. We also see the best results are achieved when $min\|lcs\|$ are set at 4, but just as mentioned in the Portuguese rule evaluation, this also delivers the smallest number of rules, which means we might be losing other rules that should be valid and that would be used by the system later. That is why we have decided to continue into the last step of the evaluation only with the possibilities of the minimum lcs being 3 or 4, rejecting the option where it is 2.

5.2.4 Analogy and Score Evaluation

When talking about the analogy module, we mentioned that each analogy would have a score and an output word that is the possible translation of the unknown word. If two analogies result in the same output word then the score of that word is the sum of the scores of both those analogies. In the end we need to sum all the scores that correspond to a certain output word to have its final score.

5.2.4.1 Calculating the Score

To tune the scores we ran the analogy module over 10 Portuguese unknown words, as explained in Section 5.2.1. The system before the evaluation had $min\|lcs\|$ set as 3 and $max\|\overline{lcs}\|$ set as 6, so with those variables unchanging, we decided to figure out what characteristics of the words we could use to calculate the score.

Before explaining the scores, we need to remember that analogies are expressed as [A : B = C : D], where A is the unknown word and (*A*, *B*) and (*C*, *D*) are two word pairs associated to the same rule; that analogy is then translated to [A' : B' = C' : D'], where A' is the proposed translation for A and (*A'*, *B'*) and (*C'*, *D'*) are two word pairs associated to the same rule.

In the beginning we only had one factor that influenced the score (f_1): the number of equal characters between words B' and D', starting from the end if the rule applied is a suffix rule or from the beginning if it is a prefix rule. To calculate the score using only this factor, we used the following equation: $score = 2^{f_1}$.

For example, if we use the analogy [apoiaram : apoiou = incorporaram : incorporou], we would get the translations as [supported : supported = incorporated : incorporated]; calculating f_1, we get that the last 3 characters of the words "*supported*" (word B') and "*incorporated*" (word D') are both "*ted*", therefore $f_1 = 3$ and $score = 8$. The results of the first evaluation, using only that factor can be seen in Table 23.

Unknown Word	Best Scored Translation(s)	Score (%)
aprazar-me-ia	-	-
excelentíssimo	-	-
interrupção	**interruption**	96%
acolhida	**welcome**	55%
	welcomes	32%
ajudam	**help**	34%
	helpe	14%
alegro	**rejoice**	58%
apoiaram	**supported**	89%
assistentes	assistant	47%
	assistants	30%
bávara	-	-
cita	**quotes**	15%
	cites	12%
	quots	11%

Table 23: Analogy scores for 10 Portuguese unknown words using only f_1.

In this table we represent the top scored translation, as well as all other translations that had at most 25% less than the top scorer and at least a 10% score. The words in bold mean that they are a valid translation to that unknown words. As we can see, we have found

translation to 7 of those unknown words. However, we decided that we were going to try to make the scores of the valid translations even higher, so that there is less of a difference, for example, between the words "*help*" and "*helpe*".

To improve the scores we decided to introduce a new factor into the calculations, which was to add to the previous score the number of characters that were replaced in word B' to reach word A' (f_2). This means the higher the number of characters to remove in the rule that relates word pairs (*B'*, *A'*) and (*D'*, *C'*), the higher the score of that analogy will be. This turns the score equation into: $score = 2^{f_1} + f_2$.

Using the example of the analogy used before ([supported : supported = incorporated : incorporated]), we can see that no character is changed between B' and A', therefore $f_2 = 0$ and $score = 8$ still. The results of using the new score are shown in Table 24.

Unknown Word	Best Scored Translation(s)	Score (%)
aprazar-me-ia	-	-
excelentíssimo	-	-
interrupção	**interruption**	95%
acolhida	**welcome**	62%
ajudam	**help**	34%
	helpe	12%
alegro	**rejoice**	59%
apoiaram	**supported**	84%
assistentes	assistant	46%
	assistants	34%
bávara	-	-
cita	**quotes**	15%
	cites	13%
	quots	10%

Table 24: Analogy scores for 10 Portuguese unknown words using f_1 and f_2.

Here we can see that some of the higher scores have lowered slightly, for words such as "*interruption*" or "*supported*". We can also see some improvements in other scores, for example the word "*helpe*" has lowered its score by 2% and the word "*assistants*" is now closer to the word "*assistant*" for the translation of "*assistentes*".

After adding a factor that changes according to the rule between the words A' and B', we decided that it would also make sense to have that same factor, but for the rule in the source language, which means the rule that relates the word pairs (*A*, *B*) and (*C*, *D*). To do this we add to the previously obtained score, the number of characters that were replaced in word A to reach word B (f_3); this number is easy to find since it is simply the level of the leaf in the trie where the rule can be found. The score equation then becomes: $score = 2^{f_1} + f_2 + f_3$.

Following the example given before, where the analogy in the source language is [apoiaram

: apoiou = incorporaram : incorporou], the number of characters replaced in word A, in this case the word "*apoiaram*", to reach word B, word "*apoiou*", is 4 and so $score = 12$. The changes to the results are shown in Table 25.

Unknown Word	Best Scored Translation(s)	Score (%)
aprazar-me-ia	-	-
excelentíssimo	-	-
interrupção	**interruption**	94%
acolhida	**welcome**	64%
ajudam	**help**	33%
	helpe	10%
alegro	**rejoice**	59%
apoiaram	**supported**	75%
assistentes	assistant	44%
	assistants	35%
bávara	-	-
cita	**quotes**	15%
	cites	13%
	quote	12%
	quots	10%

Table 25: Analogy scores for 10 Portuguese unknown words using f_1, f_2 and f_3.

The same improvements seen between the Tables 23 and 24 can be seen from the previous table to this one. Another important change to note is that the top three scored translations for the word "*cita*" are now all valid, even though still with very low scores. The word "*quote*" can be considered a translation of "*cita*" if it is used in an imperative sentence, such as "*Quote Shakespeare.*".

After looking at these scores, we can see good results all throughout, for the exception of the word "*assistentes*" having the correct word ranked second and the word "*cita*" having very low scores. The problem of the word "*assistentes*" is not easy to go around, however, with the word "*cita*", we can see that what is dragging the scores of valid translations down is the scores of various words that do not even exist in the English dictionary, such as "*quots*" or "*cits*". Therefore, one way to fix this situation is to add a fourth score. The fourth factor (f_4) multiplies the current score of the translation by 2 if the proposed word exists in the lexicon and leaves the score as it is if the word does not exist (the equivalent to multiplying the score by 1). This comes from the assumption that, if the resulting word is a known word, then we can be sure that at least it will make sense, while if the translation of the unknown word is another unknown word, it can be a word that does not even exist in the target language. The final equation for the score is: $score = (2^{f_1} + f_2 + f_3) * f_4$.

Looking back at the example we have been following, the source language analogy

[apoiaram : apoiou = incorporaram : incorporou] turns into [supported : supported = incorporated : incorporated] in the target language, suggesting the word "*supported*" as a translation. Looking at the original lexicon, the word "*supported*" already exists, as translated of the words "*apoiada*" or "*apoiado*". We can then conclude that $f_4 = 2$ and the final score is 24. The scores of the 10 unknown Portuguese words are in Table 26.

Unknown Word	Best Scored Translation(s)	Score (%)
aprazar-me-ia	-	-
excelentíssimo	-	-
interrupção	**interruption**	93%
acolhida	**welcome**	69%
ajudam	**help**	37%
alegro	**rejoice**	74%
apoiaram	**supported**	82%
assistentes	assistant **assistants**	59% 23%
bávara	-	-
cita	**quotes** **quote**	20% 16%

Table 26: Analogy scores for 10 Portuguese unknown words using all the factors.

From these results, where we use the final calculation for the scores, it is possible to see that only three words are left without any translation suggestion: "*aprazar-me-ia*", "*excelentíssimo*" and "*bávara*". In Section 5.2.5, we will have a look at what made some words not have any proposed translation, but the most puzzling out of these three words was "*excelentíssimo*".

We went through the original lexicon to try and find if there were any words ending with "*íssimo*" and found the words "*muitíssimo*", "*importantíssimo*" and "*pequeníssimo*". These words could be related to the words "*muito*", "*importante*" and "*pequeno*", respectively, to create the rules "|íssimo\o" and "|íssimo\e". The word "*excelentíssimo*" would be correctly related to the word "*excelente*" by the rule "|íssimo\e", however, since this rule only has one word pair associated with it, it is not counted as a valid rule and only "|íssimo\o" exists. This is the reason why "*excelentíssimo*" is left without translation.

In an attempt to solve the problem shown for word "*excelentíssimo*", we developed an extra step in the analogy phase, which only occurs if the word has no results in the normal analogy run. After the normal procedure of calculating analogies, if the word has no possible translation, we do all the steps again, but this time, when checking if a word exists in the lexicon we ignore the last character of that word if it is a vowel. So, for example, using the word "*excelentíssimo*" and the rule "|íssimo\o" to help us understand how this works, applying the rule to the word we reach a possible word pair with "*excelentíssimo*" and "*excelento*". Seeing

as the word "*excelento*" does not exist in the lexicon, this rule would not be used to reach a possible translation. However, if we use the developed addition of ignoring vowels, we look if any of the words "*excelenta*", "*excelente*", "*excelenti*", "*excelentu*" or "*excelenty*" (the "*y*" is sometimes considered a vowel in the English language) can be found in the lexicon. In fact, the word "*excelente*" does exist and thus we can create the analogy [excelentíssimo : excelente = muitíssimo : muito].

Using this change will not affect the results of the words that already have translations, but with it, we obtained some possible translations for the word "*excelentíssimo*", having a 33% score for each of the words "*excellent*", "*superb*" and "*splendid*". Neither one of these is the perfect translation of "*excelentíssimo*", but still they give a better sense of the translation than if there was no suggestion.

5.2.4.2 Choosing the values for $min\|lcs\|$ and $max\|\overline{lcs}\|$

The previous section was all evaluated assuming the values of $min\|lcs\|$ as 3 and $max\|\overline{lcs}\|$ as 6, but we still have not run the same evaluations for all the other possible values of these variables. In this section we intend to find out which value to give to $min\|lcs\|$ (either 3 or 4) and to $max\|\overline{lcs}\|$ (between 3 and 6).

To start this evaluation, we will use the 4 English unknown words mentioned in the evaluation setup, this will also allow us to see if the results obtained with the Portuguese words maintain when we switch to English. Table 27 shows the results, representing any word that achieved a score of over 10%.

	$min\|lcs\|$	3	3	3	3	4	4	4	4
	$max\|\overline{lcs}\|$	3	4	5	6	3	4	5	6
Unknown Word	Possible Translation(s)	Score (%)							
adjournment	**adiamento**	-	88%	82%	76%	-	89%	85%	80%
assistants	**assistentes**	66%	64%	64%	64%	67%	64%	65%	64%
	assistente	18%	18%	16%	16%	19%	18%	17%	17%
bavarian	-	-	-	-	-	-	-	-	-
beer-tasting	-	-	-	-	-	-	-	-	-

Table 27: Analogy scores for 4 English unknown words using all the factors and varying $min\|lcs\|$ and $max\|\overline{lcs}\|$.

By looking at this table we can see the results do not vary much by changing the $min\|lcs\|$ or the $max\|\overline{lcs}\|$, but we can see that in the case where $max\|\overline{lcs}\|$ is 3, the word "*adjournment*" appears without translation. This is due to the fact that the suffix "*ment*" (the $\|\overline{lcs}\|$) is 4 characters long and thus it is longer than the allowed maximum number of characters for a suffix. However, the suffix "*ment*" is largely used in the English language and so our system should be able to allow such suffix. We come to the conclusion then that $max\|\overline{lcs}\| = 3$ cannot be used in order for the system to achieve the best results. With this in mind we ran

the same tests for those 10 Portuguese words to see the results of varying $min\|lcs\|$ and $max\|\overline{lcs}\|$. Such results are in Table 28.

$min\|lcs\|$		3	3	3	4	4	4
$max\|\overline{lcs}\|$		4	5	6	4	5	6
Unknown Word	Possible Translation(s)	Score (%)					
aprazar-me-ia	-	-	-	-	-	-	-
excelentíssimo	excellent	-	-	33%	-	-	33%
	superb	-	-	33%	-	-	33%
	splendid	-	-	33%	-	-	33%
interrupção	**interruption**	94%	93%	93%	93%	93%	94%
acolhida	**welcome**	70%	69%	69%	70%	69%	69%
	welcomes	21%	21%	21%	21%	21%	21%
ajudam	**help**	35%	36%	37%	35%	35%	35%
alegro	**rejoice**	75%	74%	74%	71%	71%	71%
apoiaram	**supported**	66%	71%	82%	67%	68%	68%
assistentes	assistant	60%	58%	59%	60%	58%	58%
	assistants	31%	29%	23%	31%	29%	28%
bávara	-	-	-	-	-	-	-
cita	**quotes**	21%	22%	20%	22%	25%	25%
	quote	21%	18%	16%	22%	20%	20%
	cites	11%	9%	9%	12%	10%	11%

Table 28: Analogy scores for 10 Portuguese unknown words using all the factors and varying $min\|lcs\|$ and $max\|\overline{lcs}\|$.

Once again, the results have only a few percent differences among each other, except with the already analyzed word "*excelentíssimo*". In this case, it only returns any possible translation in the cases where $max\|\overline{lcs}\|$ is 6, this is because in the other cases, the suffix "*íssimo*" is ignored for being 6 characters long. Since this is a valid suffix and the results of the remaining words do not get that much worse by making this choice, we have decided that the $max\|\overline{lcs}\|$ should always be set at 6. As for the value in $min\|lcs\|$, since the results do not have a big difference between them and the time it takes to create the analogy of each word is very small independent of the values of the variables, we decided that it would be best to leave the variables as less strict as possible in order to allow more rules to be considered. In the end, we have opted for $min\|lcs\|$ being 3.

By testing the remaining 34 Portuguese unknown words, using the values obtained with the development test, we can see that 20 out of the 34 (58.8%) have a valid translation as its top scored word, while 3 other (8.8%) have a valid translation somewhere among its results. Only 8 of these unknown words (23.5%) were left without even a possible translation. The remaining 3 words, which have all invalid translations in the results, can still give us a perspective on what that word means. Since the output of the module is a ranked list of

translations, the best way to evaluate the results is by using the Mean Reciprocal Rank (MRR) (Voorhees [1999]). The MRR is mostly used in question answering, but by using Equation 1 on the ranked lists we obtained, we have an MRR of 0.63.

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{rank_i} \tag{1}$$

Looking at the far fewer examples with English unknown words, we have 3 out of the remaining 5 (60.0%) with a valid translation as its top scorer. Even though the test sample was much smaller than the Portuguese one, the resulting percentage of correct translations is still the same, giving us a good perspective on the overall results of this module. Since the remaining 2 words either have no possible translations or no valid translation, the MRR is 0.60.

We have also tested this module in a real life scenario. We gathered the words that were left untranslated from Moses [9] (a globally used statistical machine translation system) and tried to use this analogy module to obtain results. Out of 101 words, 36 have a correct translation as their first suggestion and the MRR is 0.395. The results are lower than the ones tested before because these English words were extracted from a different source, when comparing to the lexicon that was extracted from Europarl corpora. On the other hand, the Portuguese unknown words were also extracted from Europarl, being then more likely to have related words in the lexicon.

5.2.5 Error Analysis

After evaluating the system, there are various words that were left without translation or with only wrong translations as suggestions. With Portuguese words, there were a total of 10 words left without translation, where 5 of them have a hyphen ("-"), which can be a possible reason as to why those words had no translation, since hyphens add complexity to Portuguese words. Other words without translation are the ones referring to countries or areas, such as "bávara" or "luxemburgueses". This is mainly due to the fact that these are places that are not mentioned frequently and therefore do not have entries in the lexicon. The remaining 3 words are "degustação", "pátio" and "rectificar"; while "pátio" is a word without any related words, "degustação" and "rectificar" simply do not have any similar words in the lexicon.

On the other side of errors, there are words that have suggested translations, but these suggestions are not correct. In the Portuguese results we can find 4 of these: "excelentíssimo", "desejar-vos", "entender-nos" and "estive". Two of these have the same hyphen problem we mentioned in the previous paragraph; however, if we look at their suggested translations they can still give us a hint on what the word means, especially the word "desejar-vos", which has "wishe" as a suggestion and, even though that word does not exist,

[9] http://www.statmt.org/moses/

it can point us in the direction that “wish” is a possible translation. The word “excelentíssimo” has a special meaning in Portuguese, that is not related to the word “excellent”. When used in the context of “excelentíssimo senhor”, having the translation “excellent sir” is not correct, but does not deprive the sentence of its meaning either. Finally the word “estive” is a complicated word to translate in any way that is not through a lexicon, since the verb “to be” has conjugations that are not similar in the way they are written. For instance, “eu estou” is translated to “I am” while “eu estive” is translated to “I was”. This difference in conjugation makes this verb hard to translate through analogy.

When testing the 9 Europarl English unknown words we were left with 3 without translation and 1 with an incorrect translation. The errors are very similar to the ones seen in Portuguese: hyphens (“beer-tasting”), words related to places (“bavarian”) and words that have few other related words (“merry” and “courtyard”). Looking at the far larger example of words extracted with Moses, we see that most o the words left with no translation are nouns (“lounge”, “teapot” or “janitor”); verbs are far easier to translate, given that some conjugation of the verb might be in the lexicon, while the only changes a noun can have in English are singular or plural.

5.3 Context Module

The context module bases itself on the words that surround an unknown word. To evaluate it, we need to create a context for each word and see how two contexts from words in different languages relate to each other. If the two words have similar meanings, then their respective contexts should be similar and, with this, we can extract possible translations of unknown words.

5.3.1 Evaluation Setup

Just as the analogy module, we use Europarl parallel texts to evaluate the context module. The same bilingual lexicon is used, as well as the same Portuguese and English corpora, both having one text with 10000 lines and one with 100 lines. Both of these sets of corpora are used in the testing phase, since there is no need for a training phase in this module.

5.3.2 Context Evaluation

We extracted all the unique words from each of these corpora and their respective context; as explained in Section 4.3. The context of a word A is words L1, L2, R1 and R2, where L1 and L2 are the first two key words (a key word is an adjective, an adverb, a name or a verb) to the left of word A and R1 and R2 are the first two key words to the right of word A.

In Tables 29 and 30, we list some examples of words and respective context words that were extracted in the way previously mentioned. Note that along with these words, many

other invalid context words were found associated to the listed words.

word	valid context word
gentlemen	ladies
aid	financial countries package
report	final payment
car	manufacturers
farming	farmers

Table 29: English examples of words and their valid context words.

word	valid context word
senhoras	senhores
lugar	aqui
gregos	italianos
deputado	colega senhor
termino	final

Table 30: Portuguese examples of words and their valid context words.

When comparing the context lists of the 100-line corpus with the 10000-line corpus, we can see that the 10000-line is more accurate. This is due to a larger sample of context words extracted per word, since a word is more likely to appear more than once. This will generate a bigger context list, which might contain more words that are not helpful, but also has a higher probability of including words that do help us in determining a translation.

After extracting the context, we tried to get a translation of all the words, by looking them up in the bilingual lexicon; if the word did not have a translation, therefore considered an unknown word, we would compare its own context to the context of all the words in the opposite language. If the context is similar, those two words could be translations of each other. After calculating all the context's similarities, the system outputs a list of possible translations, ranked by probability of being a correct translation.

In general, the results of this module were worse than expected. We started by evaluating the 100-line corpora, translating words from Portuguese to English.

In total there were 31 unknown words found in the Portuguese corpus and, out of these, only 2 had no suggested translation; this is a good result, however, out of the remaining 28, 18 did not have any correct translation amongst its results. So, in the end, only 11 words had at least one suggestion that was related to the unknown word. In Table 31, we can see that

many of the translations were among the highest ranked words suggested as translation, if we use Equation 1 to figure out a value for MRR, we reach a result of 0.28. However, considering there are many tied words, we can calculate the MRR with the lowest rank that word could achieve; for instance, a word ranked first but tied with 15 other words would have its lowest possible rank as 16. Using these ranks, the MRR would then be only 0.1, which is a very low result.

Unknown word	Valid translation	Size of output	Rank
compreenderiam	understanding	16	1 (tied with 15 other words)
degradado	deteriorated	23	1 (tied with 1 other word)
luxemburgueses	luxembourg	15	2 (tied with 13 other words)
mayer	mayer	3	1 (tied with 1 other word)
perturba	disturbing	6	1 (tied with 5 other words)
rectificar	rectify	9	1
reduzirem	reducing	16	1 (tied with 15 other words)
reduzisse	reducing	15	1 (tied with 14 other words)
reformados	pensioners	29	11 (tied with 13 other words)
representamos	represent	10	1 (tied with 1 other word)
visa	aims	31	5 (tied with 26 other words)

Table 31: Portuguese unknown words and their valid translations using the context module.

If we use the same 100-line corpora, but translate the unknown words in English, the system only finds 12 words to translate; out of those, 1 has no suggested translation and 9 have no valid translations, leaving only 2 words with valid translations. These are shown in Table 32. The MRR is 0.11 and, when using the lowest possible ranks, it goes down to 0.05.

Unknown word	Valid translation	Size of output	Rank
aims	objectivo	16	3 (tied with 13 other words)
	visa	16	3 (tied with 13 other words)
mayer	mayer	7	1 (tied with 1 other word)

Table 32: English unknown words and their valid translations using the context module.

Since the number of words is limited, a word appears in average only one time in the whole text. This makes the context list not very accurate and, therefore, outputs a lot of possible translations that, in fact, only have one word in common with the unknown word. The existence of some verbs that appear very often in sentences, such as “am” or “is”, can also lead to the construction of context lists that do not depict the true nature of the unknown word. The high frequency of these verbs makes many words seem related to them, when, in fact, those verbs do not tell us anything about the word itself.

We then evaluated the 10000-word corpora, translating from Portuguese to English. The

results were expected to be higher than the previous ones, given that words have a higher chance of appearing more than once, which will make the context more accurate. However, in the Portuguese to English evaluation, we see a decrease in the MRR, going down to 0.06 with the best rank the word can achieve and 0.02 if we calculate it with the worst rank. One possible explanation is that the context is so big that hundreds of words are associated to that unknown word, but also many of the unknown words were verbs that do not have a specific context associated to them, such as “obterem” or “entendi”, which makes it harder to find a translation through context.

Evaluating the same 10000-word corpora, translating from English to Portuguese, we obtain slightly better results, as expected, reaching an MRR score of 0.22 in the best case and 0.12 in the worst case scenario. These results are more in tune with what we expected would happen, because the unknown words consist more of nouns rather than inflections of verbs.

5.4 Evaluation of the Whole System

To evaluate the system all together, we used the offline method explained in Section 4.4.3. The lexicon and the unknown words used were the same that were used in the Analogy module evaluation, to guarantee that we could compare these results to the results of using of analogy by itself.

Unknown Word	Possible Translation(s)	Analogy Score	Cognate Score
aprazar-me-ia	-	-	-
excelentíssimo	-	-	-
interrupção	interruption	93%	-
acolhida	welcome	69%	-
ajudam	help	37%	-
alegro	rejoice	74%	-
apoiaram	supported	82%	-
assistentes	assistants	23%	-
bávara	-	-	-
cita	quotes	20%	-
	quote	16%	-
	cites	9%	-
	cites	-	64%

Table 33: Analogy and Cognate scores for 10 Portuguese unknown words.

The context module was not included in the merged system due to its poor results, as shown in the previous section. Considering it needs extra data (the context lists) as well as extra uploads by the user (corpora with the unknown words so the context can be ex-

tracted), we felt that its results were not good enough to compensate making the system more complex.

As a first step, we used 10 Portuguese words to compare the results of using the analogy module and the cognate module by themselves. The results of the valid words can be seen in Table 33.

As it is clearly visible, the results of analogy are far superior to those of the cognate module by itself. However, the cognate module did encounter one word that the analogy module did not, which can be beneficial to the system since this might be the translation the user was looking for. Also, part of the reason some of the words have a score of zero, is because these words never appear in the lexicon and, thus, the cognate module will never evaluate them. When merging the two systems together we decided that all the translations suggested by the analogy module would enter the cognate module in an attempt to obtain a cognate score. This was tested in the first evaluation of the merged system, where both the analogy and the cognate module count for 50% of the final scores. Table 34 shows the results.

Unknown Word	Possible Translation(s)	Score
aprazar-me-ia	-	-
excelentíssimo	-	-
interrupção	interruption	96%
acolhida	welcome	34%
ajudam	help	18%
alegro	rejoice	37%
apoiaram	supported	41%
assistentes	assistants	49%
bávara	-	-
cita	quotes	10%
	quote	8%
	cites	12%
	cites	32%

Table 34: Score attributed by the system to 10 Portuguese unknown words, using the analogy module and the cognate module with weights 1.

The usage of the two modules at once has raised the score of some words, such as “interruption” or “assistants”, due to the similarity these words have with the unknown word. In this sense, the cognate module can be of some help in calculating the possible translations. This merge has also lowered the scores of any translation that is not similar to the unknown word. To reach a balance between these scores, we decided to test different weights to the analogy module, maintaining the cognate module’s weight at 1, in order to raise the scores of the translations that are not similar to the unknown word.

		Analogy Weight			
Unknown Word	Possible Translation(s)	1	2	3	4
aprazar-me-ia	-	-	-	-	-
excelentíssimo	-	-	-	-	-
interrupção	interruption	96%	95%	94%	94%
acolhida	welcome	34%	46%	51%	55%
ajudam	help	18%	24%	27%	28%
alegro	rejoice	37%	50%	56%	60%
apoiaram	supported	41%	54%	61%	65%
assistentes	assistants	49%	40%	36%	33%
bávara	-	-	-	-	-
cita	quotes	10%	13%	15%	16%
	quote	8%	10%	12%	12%
	cites	12%	11%	10%	10%
	cites	32%	22%	16%	13%

Table 35: Score attributed by the system to 10 Portuguese unknown words, varying the analogy module's weight.

In Table 35, we can see that the results evolve as expected; the translations that are similar to the unknown word decrease their score and the ones that have no similarities increase it as the analogy weight rises.

The final choice of what weights to use is up to the user, but if this option is left blank in the offline mode, or if the weights are both left at 0 in the online mode, the system will choose a default value for both weights. To decide which are the most valid weights, we wanted to select the highest analogy weight that would maximize the MRR. By calculating the MRR for all the 34 Portuguese unknown words that we have used in the analogy module, we conclude that in case the analogy weight is 1, 2 or 3, the MRR is 0.70; while if the analogy weight is 4 the MRR decreases to 0.69. Both these results are considerably higher than using the analogy module by itself and thus the default values for the analogy weight and the cognate weight are 3 and 1, respectively.

The main words that cause an increase in the MRR, when compared to the analogy module, are:

- "congratula-se": the word has no translation with the analogy module, but using the merged module, it correctly guesses "congragulates" as the most likely translation;
- "pátio", similarly: the system only finds the translation "patio" when using the cognate module;
- "rectificar": even though "rectify" is ranked as the second most likely translation, it is still an improvement compared to the no valid translations with the analogy module.

On the other hand, when we set the analogy weight at 4, the word “ignores” goes from top ranked to second place, as a translation to the word “ignoro”. That is the reason why we decided that the default value of the weight would be 3. The full tables with all the results from Portuguese unknown words can be found in Appendix C.

When testing with the real life scenario explained at the end of Section 5.2.4.1, we can also slightly improve the MRR. Instead of being 0.395, we reach 0.417 with the default values we have chosen. This is not such a clear increase as with the Portuguese evaluation, but the words “fiber”, “liters” and “mortuary” now have valid translations, while when using solely the analogy module, the system did not return any suggested translation.

5.5 Summary

We started evaluating the system by adding the Part of Speech tagger to the Cognate Module. This resulted in a 4.8% increase in Precision, which means a lower number of wrongly marked cognates. We also tested the Named Entity Translation system created by Luís Carvalho, realizing that only 1.6% of the Named Entities were correctly translated after using the system.

The Analogy Module obtained the best results, with the maximum result obtained being an MRR of 0.60 for the evaluated Portuguese Unknown words and 0.395 for the English Unknown Words extracted from an SMT. The most common errors were encountered in words with hyphens, words referring to geographical areas and words with few related words.

The results of the Context Module were a lot lower than expected, reaching only an MRR of 0.28 as the best possible score.

For that reason, the final system only uses Analogy and Cognates, having the best result when the Analogy comprises 75% of the system, finishing with an MRR of 0.70 for the Portuguese unknown words and 0.417 for the English unknown words.

6 Conclusion and Future Work

6.1 Resume

SMT systems have to deal with unknown words, that is, words that were not seen during training. Thus, having a system that proposes translations to these unknown words can improve SMT systems' results.

One of the ways to find translations of unknown words is to find possible translations to these words by using cognates. If two words are considered to be cognates, there is a strong possibility that they are translations of each other. In the framework described in this paper, we used a set of similarity measures to determine if two words are cognates. However, this is not an easy task, since there are a number of false cognates and also because many words that are translation of each other are not cognates. The cognate detection using a POS Tagger, manages to correctly determine 55% of the total cognates that exist in a parallel corpora, however it also assumes as cognates many other words that are not. Thus, we have implemented a module that follows the Logical Analogy paradigm in order to find possible translations of unknown words.

The analogy module makes use of the fact that the unknown word is probably similar to some known words and therefore we can deduce a possible translation from that. This module proves to be efficient for verbs, since these have various inflections. We can gather information from other inflections of the same verb in order to decide on a translation for the unknown word. This module was able to translate 66% of the Portuguese unknown words found on the same website as the words included in the used lexicon, with a Mean Reciprocal Rank (MRR) of 0.63. In order to evaluate the model in a real life scenario we extracted English unknown words from an execution of an SMT and attempted to translate them. The results were lower than when using words from the same website, finding translation to 46% of these words and resulting in an MRR of 0.395.

A third module was developed, using context to attempt to find a relation between two words. Two words that are similar, even in different languages, will likely be surrounded with other similar words. This module based itself on parallel corpora and on knowing how to translate the surrounding words, so that, in the end, the relations between the contexts could lead us into valid translations. However, the results of this module were much lower than expected, reaching a best case scenario of 0.22 as the MRR. These poor results led us to decide not to include the context module in the final system.

We then merged the cognate and the analogy modules, using certain characteristics from both and giving the analogy module a bigger weight in the results of the system. We managed to improve the results previously obtained by testing the systems individually. Using the same two scenarios, used in the analogy module, to translate Portuguese and English unknown words, the MRR rises to 0.70 when translating the Portuguese words and to 0.417 when using the English unknown words extracted from an SMT.

6.2 Contributions

These are the main contributions achieved from this work:

- Research on various methods to translate unknown words.
- Adaptation of a previously existing module to detect cognates, in order to improve its precision.
- Construction of modules that use analogy and context to find translations of unknown words.
- Merger of the modules in order to achieve the best results.
- Evaluation of the system.
- Publication of paper “Dealing with unknown words in statistical machine translation” in the LREC 2012 conference.

6.3 Future Work

The main improvement that could be done in future work has to do with the context module. The concept of the module is valid, using surrounding words to relate two words in different languages; however, the results were poor. One of the main reasons for the low results comes from the fact that the words that surround the unknown word can be words that appear too often. We had a first approach at trying to solve that problem, ignoring some classes of words such as determiners or prepositions. An extra step that could be made would be to use the frequency of a word in its score. For instance, verbs such as “is” or “does” are used more often than other verbs such as “fishes” or “drives”. This increase in frequency makes it harder to associate the verbs to a specific word since they appear in the context of most words. If we could decrease the score as the frequency increases, the results of the context module could increase.

Even though the system focuses on translations from English to Portuguese and vice-versa, it would be interesting to have a system that can be easily adapted to any other languages. The cognate module is the only part of the system that is not fully adaptable, since it needs a list of transliteration rules, which are specific to a pair of languages. To allow the system to be used by other languages we need to introduce different transliteration rules. If, instead of writing the transliteration rules, the system could calculate them before the cognate module, we could use the system with any pair of languages. To do this, we could adapt the rule extraction explained in Section 5.2.3 to be able to compare two words in different languages, instead of words in the same language. By doing this before the cognate detection, these rules can be used as transliteration rules and the system is not constrained to English and Portuguese.

Regarding the analogy module, the method chosen to deal with words that have no suggested translation was to switch the last character of a word (in case it is a vowel), by the different vowels to see if a translation could be obtained. Another possible approach would be to attempt to use rules that were previously discarded. The system initially discards every rule that does not have at least one word pair associated to it, but some of those rules are valid. Therefore, in the case there was no translation to an unknown word, we could recycle these rules to see if any of these fit the unknown word.

References

Y. Al-Onaizan and K. Knight. Translating named entities using monolingual and bilingual resources. In *in Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 400–408, 2002.

K. Arora, M. Paul, and E. Sumita. Translation of unknown words in phrase-based statistical machine translation for languages of rich morphology. In *in Proceedings of the First International Workshop on Spoken Language Technologies for Uner-Resourced Languages*, pages 70–75, 2008.

C. Callison-Burch, P. Koehn, and M. Osborne. Improved statistical machine translation using paraphrases. In *in Proceedings of HLT-NAACL*, pages 17–24, 2006.

L. Carvalho. Criação de léxicos bilingues para tradução automática estatística. Master's thesis, Instituto Superior Técnico, 2010.

L. R. Dice. Measures of the amount of ecologic association between species. *Journal of Ecology*, 26:297–302, 1945.

M. Eck, S. Vogel, and A. Waibel. Communicating unknown words in machine translation. In *in Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, 2008.

C. Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, 1998. URL `http://books.google.es/books?hl=es&lr=&id=Rehu8OOzMIMC`.

E. Fredkin. Trie memory. *Communications of the ACM*, 3(9):490–499, 1960.

P. Fung and L. Y. Yee. An ir approach for translating new words from nonparallel, comparable texts. In *in Proceedings of COLING-ACL*, pages 414–420, 1998.

A. Hassan, H. Fahmy, and H. Hassan. Improving named entity translation by exploiting comparable and parallel corpora. In *in Proceedings of the 2007 Conference on Recent Advances in Natural Language Processing (RANLP)*, pages 2–7, 2007.

F. Huang. Cluster-specific named entity translation. In *in Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, pages 435–442, 2005.

F. Huang and S. Vogel. Improved named entity translation and bilingual named entity extraction. In *in Proceedings of the 4th IEEE International Conference on Multimodal Interface*, pages 253–258, 2002.

L. Gomes J. Costa, G. P. Lopes and L. M. S. Russo. Representing a bilingual lexicon with suffix trees. In *in Proceedings of 26th Symposium On Applied Computing*, pages 1164–1165, 2011.

P. Jaccard. tude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin de la Socit Vaudoise des Sciences Naturelles*, 37:547–579, 1901.

L. Jiang, M. Zhou, L. Chien, and C. Niu. Named entity translation with web mining and transliteration. In *in Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 1629–1634, 2007.

P. Koehn and K. Knight. Learning a translation lexicon from monolingual corpora. In *in Proceedings of the Workshop of the ACL Special Interest Group on the Lexicon (SIGLEX)*, pages 9–16, 2002.

G. Kondrak. Identifying cognates by phonetic and semantic similarity. In *in Proceedings of NAACL 2001: 2nd Meeting of the North American Chapter of the Association for Computational Lingustics*, pages 103–110, 2001.

G. Kondrak, D. Marcu, and K. Knight. Cognates can improve statistical translation models. In *in Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, pages 46–48, 2003.

P. Langlais and A. Patry. Translating unknown words by analogical learning. In *in Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 877–886, 2007.

V. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *in Soviet Physics Doklady*, pages 707–710, 1966.

W. Ling. Named entity translation using anchor texts, 2010. Internal Work for the Course of IR.

G. S. Mann and d. Yarowski. Multipath translation lexicon induction via bridge languages. In *in Proceedings of NAACL*, pages 151–158, 2001.

A. Mulloni and V. Pekar. Automatic detection of ortographic cues for cognate recognition. In *in Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, pages 2387–2390, 2006.

T. Rama and L. Borin. Estimating language relationships from a parallel corpus. a study of the europarl corpus. In *in Proceedings of NODALIDA*, pages 161–167, 2011.

M. Simard, G. Foster, and P. Isabelle. Using cognates to align sentences in bilingual corpora. In *in Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI)*, pages 67–81, 1992.

E. Ukkonen. On-line construction of suffix trees. *Algorithmica*, 14(3):249–260, 1995.

E. M. Voorhees. Trec-8 question answering track report. In *in Proceedings of the 8th Text Retrieval Conference*, pages 77–82, 1999.

P. Weiner. Linear pattern matching algorithms. In *in Proceedings of the 14th Annual Symposium on Switching and Autonata Theory*, pages 1–11, 1973.

W. E. Winkler. String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage. In *in Proceedings of the Section on Survey Research Methods*, pages 354–359, 1990.

J. Zobel and P. Dart. Phonetic string matching: Lessons from information retrieval. In *in Proceedings of the 19th ACM International Conference on Information Retrieval (SIGIR'96)*, pages 166–172, 1996.

A Similarity Measures

- *Dice Coefficient*, focuses on the number of letters in common between the two words and is given by Equation 2;

$$Dice(x, y) = \frac{2|x \cap y|}{|x| + |y|} \tag{2}$$

-*Jaccard Distance* [see Jaccard, 1901] is similar to the *Dice Coefficient* and is given by Equation 3;

$$Jaccard(x, y) = \frac{|x \cap y|}{|x \cup y|} \tag{3}$$

-*Jaro Winkler* [see Winkler, 1990] is composed by the *Jaro Distance* and a prefix function. The equation of the *Jaro Distance* is given by Equation 4. In this equation, *m* is the number of matching characters between the two words. A character is considered matching if its position in the two words does not differ more than the coefficient calculated by Equation 5. In the *Jaro Distance* the *t* is the number of transpositions the matching characters need to go through to be in the same order, for example in the words *nose* and *one*, there are 3 matching characters, but they are in different orders (*noe* and *one* respectively), this means that there needs to be one transposition for the characters to be arranged in the same order and thus *t* = 1. Equation 6 is the full equation of the *Jaro Winkler* measure, adding the *Jaro Distance* to a function where *l* is the number of prefix characters the words have in common (*nose* and *one* would have *l* = 0, but *one* and *once* would have *l* = 2), to a maximum of 4 and *p* is a constant factor that is normally 0.1.

$$d_j = \frac{1}{3}(\frac{m}{|x|} + \frac{m}{|y|} + \frac{m-t}{m}) \tag{4}$$

$$\lfloor \frac{\max(|x|, |y|)}{2} \rfloor - 1 \tag{5}$$

$$d_w = d_j + (lp(1 - d_j)) \tag{6}$$

-*LCSR* is calculated as the quotient between the longest sequence of letters that is common in both words (for example, the longest common subsequence between *night* and *nacht* is *nht*) and the length of the longest word;

$$lcsr(x, y) = \frac{|lcs(x, y)|}{max(|x|, |y|)|} \tag{7}$$

-*LCSRC* is similar to *LCSM*, however it only counts consonants because two cognate words vary in vowels more commonly than they vary in consonants, so the consonants in the words are a better indicator of if the words are cognates or not. The equation is given by Equation 8;

$$lcsrc(x, y) = \frac{|lcs(cons(x), cons(y))|}{max(|cons(x)|, |cons(y)|)} \tag{8}$$

-*Identical Words*, this measure simply returns 1 if the words are spelled the same way and 0 otherwise;

$$ident\_words(x,y) = \begin{cases} 1, case x = y \\ 0, otherwise \end{cases} \tag{9}$$

-*Word Length*, considering cognates normally do not have a big difference in length, the score of this measure is closer to 1 when their length is more similar;

$$1 - \frac{max(|x|,|y|) - min(|x|,|y|)}{max(|x|,|y|)} \tag{10}$$

-*Sequence Letters* is similar to *LCSR* or *LCSRC*, but it calculates the longest sequence of characters that is common in both words and without interruption of other characters, so using the same example as before, between *night* and *nacht*, the longest sequence of characters in this case would be just *ht*;

$$s\_l(x,y) = max(|commonSequenceLetters(x,y)|) \tag{11}$$

$$\frac{s\_l(x,y)}{min(|x|,|y|)} \tag{12}$$

-*Levenshtein Distance* returns the minimum number of operations (substitutions, deletions and insertions) that need to be done in the characters of one word to turn it into another word. One common example given is to calculate the *Levenshtein Distance* of the words *Saturday* and *Sunday*, which is 3 because to convert one word into the other we need to do 3 operations, for example remove the second *a* and the *t* of *Saturday* to turn it into *Surday* and then replace the letter *r* by *n* to achieve the target word *Sunday*. Since this measure does not return a value in [0,1], it needed to be normalized, as shown in Equation 13;

$$\frac{max(|x|,|y|) - lev(x,y)}{max(|x|,|y|)} \tag{13}$$

-*Soundex* [10] is a measure that takes into consideration the sound or phonetic of words. Soundex makes use of the fact that some letters sound similar to determine whether two words could be cognates or not. It has limitations though, such as the fact that it is only meant for single-letter sounds and that it is meant for English words. The measure returns 1 if the sound of the words is the same and 0 otherwise. As an example, we can go back into using the words *night* and *nacht*. According to the rules of *Soundex*, the vowels and the *h* should be ignored, so the words would change to *ngt* and *nct*. Since *g* and *c* are considered to sound phonetically similar by *Soundex*, both words would output the same code N-230 (first letter N, the sound of *c* and *g* has code 2 and the sound of *t* has code 3);

$$soundex(x,y) = \begin{cases} 1, case soundex(x) = soundex(y) \\ 0, otherwise \end{cases} \tag{14}$$

[10] http://www.archives.gov/research/census/soundex.html

B Conversions from English tagger to Portuguese tagger

English Tag	Portuguese Tag	Lexical Category
JJ JJR JJS	ADJ	Adjective
RB RBR RBS WRB	ADV	Adverb
DT WDT	DET	Determiner
CD	CARD	Cardinal
NN NNS NP NPS	NOM	Name
PP PP$ WP WP$	P	Pronoun
CC RP IN TO	PREP	Preposition
VBG MV VB VBZ VBP VBN VBD	V	Verb
UH	I	Interjection
PDT	NOM and ADJ	Name and Adjective

Table 36: Conversion from the English tagger's categories to the Portuguese tagger's categories.

C Full Results for Translation of Portuguese Unknown Words

Unknown Word	Best Scored Translation(s)	Score (%)
aprazar-me-ia	-	-
excelentíssimo	excellent	12%
	superb	12%
	splendid	12%
interrupção	**interruption**	94%
acolhida	**welcome**	51%
	welcomes	15%
ajudam	**help**	27%
alegro	**rejoice**	56%
apoiaram	**supported**	61%
assistentes	assistant	55%
	assistants	36%
	assistent	16%
bávara	-	-
cita	**cite**	16%
	quotes	15%
	quote	12%
colocaria	**put**	15%
	place	13%
compreenderiam	**understand**	71%
concedesse	**grant**	28%
	bestow	16%
	concede	11%
confirmará	**confirm**	44%
	confirme	20%
	confirmate	14%
congratula-se	**congratulates**	11%
constar	**contained**	29%
	contain	15%
	included	12%
	constate	12%
	const	11%

Table 37: Analogy scores for of the Portuguese unknown words (translations with score of more than 10% or top scored words shown). [Part 1/3]

Unknown Word	Best Scored Translation(s)	Score (%)
contactar	**contact**	47%
	touch	22%
	contactary	18%
	contacte	13%
	contacting	12%
	contacts	12%
	contacted	12%
debruçarem	**address**	38%
	looked	11%
degradado	**degraded**	73%
	degradated	21%
	degradate	16%
degustação	-	-
desejar-vos	wishe	75%
dirigiu	**directed**	62%
disponibilizada	**available**	40%
	provided	19%
dívidas	debt	46%
	debts	24%
entender-nos	hinted	28%
	deduce	35%
estive	iss	32%
figuro	**figure**	43%
	figures	19%
	personification	19%
	diagrams	11%
ignoro	**ignore**	41%
	ignores	39%
	ignor	24%

Table 38: Analogy scores for of the Portuguese unknown words (translations with score of more than 10% or top scored words shown). [Part 2/3]

Unknown Word	Best Scored Translation(s)	Score (%)
incluíam	**included**	71%
incumbisse	incumbent	33%
	instruct	21%
	instructed	13%
	responsible	11%
integram	**integrate**	47%
	integrat	21%
	integra	19%
	integrates	16%
	integral	14%
	integre	13%
	integr	12%
	integry	11%
linguísticos	**linguistics**	55%
	linguistic	42%
	linguistical	15%
luxemburgueses	-	-
mal-entendido	-	-
pedir-lhe-ei	-	-
perturba	**disturbs**	21%
pronunciar-me-ei	-	-
pátio	**patio**	25%
	pat	11%
rectificar	-	-
recusasse	**refuse**	32%
reduzirem	**reduce**	47%
reduzisse	**reduce**	22%
	reduced	12%
	reduc	12%
	reducing	12%
reformados	**reformed**	61%
	reformeds	19%
representamos	**represent**	60%
	represents	16%
	representations	11%
	representate	11%

Table 39: Analogy scores for of the Portuguese unknown words (translations with score of more than 10% or top scored words shown). [Part 3/3]