Building an Anglo-Spanish Translation Lexicon for Proper Names form the Web

Ivan Lyubomirov Pavlov

Supervisor: Dr. Robert Gaizauskas

COM3010

November 2009 – January 2010

Signed Declaration

All sentences or passages quoted in this dissertation from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations which are not the work of the author of this dissertation have been used with the explicit permission of the originator (where possible) and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this dissertation and the degree examination as a whole.

Name: Ivan Lyubomirov Pavlov

Signature:

Date: 18/01/2010

Abstract

The use of electronic dictionaries is common in tasks like Cross Language Information Retrieval (CLIR) and Machine Translation (MT). There are many electronic dictionaries, but they do not include proper names. Furthermore, new names appear every day. Since names are an essential part in CLIR queries and in language in general, there is a need for a means to translate them from one language to another. The aim of this project has been to construct a translation lexicon for proper names obtaining the data from translated texts from the Web. Ideally, this lexicon would update continuously from the Web's growing resources.

This report contains a literature research and an analysis of the findings which has finally led to the design and development of a system that constructs a translation lexicon for proper names, from comparable news articles in English and Spanish downloaded from the Web. The lexicon contains almost 10,000 named entities recovered from more than 27,000 pairs of articles. It has the capability of updating its contents from new articles appearing every day on the Web.

Table of Contents

1. Introduction

1.1. Background

Searching the Web has become such a necessity that *to google* is commonly used as a verb. But people's search potential is still limited by factors such as language. In a perfect scenario the query would be written in one language and the results would be relevant documents in any language. Since MT is very advanced nowadays, it is not necessary to know the source language of the results because they can be translated automatically. In order to make this scenario real, there is a lot of research on CLIR and MT. Those two techniques rely on bilingual dictionaries for translations. However, proper nouns such as person and organization names are not included in those dictionaries. Nevertheless, names are quite often an essential part of a query or even the whole query. Normally, names tend to be the same in different languages, especially when they use practically the same alphabet as English and Spanish, but it is not always the case. Location names sometimes differ, for example London is Londres in Spanish. The same happens to person names, e.g. Prince Charles is called Carlos in Spain. Another example is movie names, which could be considered normal text, but even human translators have difficulties with them because movies are rarely translated literary, so the translator needs to know how the movie is called in the foreign country. Furthermore, new names are constantly introduced making it very difficult to include them in a named entities dictionary.

1.2. Objectives

The primary objective of this project is to build an Anglo-Spanish translation lexicon for proper names. This lexicon should be constructed automatically from bilingual texts obtained from the Web. A secondary objective is that the lexicon is able to update continually with new names that appear in the Web on a regular basis. The main constraint of the project is time as it is developed in one semester only.

In order to achieve the objectives, a system must be designed that locates and downloads translated web pages in English and Spanish constructing a parallel corpus. Then the system should locate the named entities and align them obtaining pairs of names in English and their translations in Spanish.

There are three clear parts in the project, and various alternatives exist for solving each one of them. The first part is obtaining an English-Spanish parallel corpus. There are different options for this such as taking an already existing corpus form the web or obtaining it by the use of a web crawler capable of identifying the translated texts. An alternative to using translated texts is a comparable corpora consisting of documents which are not translations but have the same topic. For example a lot of documents with similar information, written in different languages, are uploaded daily from news agencies.

The second part of the project consists in recognizing what is a proper name in a text, both in English and Spanish. The alternatives here are to use simple heuristic rules or more complicated techniques involving machine learning. There are also various free and open source software tools which might be of help.

The last part is to match the names in one language to the ones in the other. Having localized the named entities, the texts have to be aligned in order to pair them. Some heuristics or statistical models might be used for achieving this final task.

1.3. Report Structure

Chapter 2 reviews relevant literature and other work on the same or similar topics. The aim is to investigate the alternatives for completing the objectives.

Chapter 3 briefly presents the requirements and then analyses the different alternatives for meeting them. The decision for the most appropriate chose is discussed in this chapter.

Chapter 4 describes the design and implementation of the system.

Chapter 5 describes the proceedings for evaluating the different parts of the system and discusses the results.

Chapter 6 discusses what could be done from now on in order to improve the system's performance and results.

Chapter 7 makes conclusions on the overall work accomplished.

Chapter 8 contains the list of references used throughout this report.

2. Literature Research

This section discusses relevant literature and other projects on the same or similar topics. The objective is to review different alternatives for completing the project objectives. The section is divided in three parts, one for each part of the project.

2.1. Construction of a Bilingual Corpus

The first step in the project is to obtain a lot of aligned texts in English and Spanish containing named entities. There are a lot of works on this topic and therefore a lot of options to choose from.

2.1.1. STRAND

One of the most cited authors, on the topic of mining the web for obtaining translated pages, is Philip Resnik. His work, the STRAND (Structural Translation Recognition for Acquiring Natural Data) system, is described in (1). STRAND is a system that automatically acquires pairs of translated texts from the Web. It is composed of two basic parts. The first one is obtaining the candidate pairs, and the second one is classifying them as valid or not. The report on the system describes how an English-French parallel corpus is constructed.

Classical STRAND uses AltaVista search engine advanced search to gather candidate pairs of translations. Basically it uses two different approaches. The first is to search for pages in any language where there are links named "English" or "anglais" and "French" or "francais". If the links are close enough in the text, the pages are considered a candidate pair.

The second way is to look for pages in French with a link called "English" or "anglais". And vice versa, pages in English with links called "French" or "francais".

Another way to gather candidate pairs is by downloading the entire site if it contains pages in the two target languages and making the cross product of the pages. To obtain the candidate pairs some URL-matching is performed based on the assumption that directory structure of the web site might be the same in different languages. Another way for matching the pairs is by content length, using a constant tuned for the language pair.

After the candidate pairs are generated, structural filtering is performed. The HTML is linearized and converted in three types of tokens: one denoting the beginning of an html tag, one denoting the length of the tag and one denoting the end. The two pages are compared based on statistics on those tokens.

The reported results of STRAND are almost 100% precision and more than 60% recall.

The report then describes some improvements of the system using decision trees and bilingual dictionaries.

The last improvement of STRAND modifies it to use the Internet Archive (http://www.archive.org/). The Internet Archive is a nonprofit organization which tries to store copies of every public Web site from different points in time.

Instead of generating candidate translation pairs using AltaVista, STRAND uses the Internet Archive. It craws the stored pages and generates candidate translation pairs by URL matching. The algorithm for matching removes any parts of the URL that might designate the language of the page and pairs the URLs which give the same result. The Internet Archive already gives the language of the page so there is no need for language recognition.

Resnik's Web page provides several bilingual databases generated with STRAND but none of them is Anglo-Spanish.

2.1.2. BITS

Another interesting and very popular work is (2). This report describes another system for generating parallel corpora from the Web called BITS.

BITS uses a simple algorithm. It generates a list of candidate pairs by querying a DNS server for sites in the domain of the minor of the languages of interest. In the paper this is the .de domain for German.

The sites are then identified as monolingual or bilingual by using a language identifier on the pages of the top 3 or 4 levels. If there are pages in more than one language in those levels the whole site is downloaded using GNU Wget. Only the plain HTML files are obtained, leaving out the pictures. Then HTML is converted to plain text. Pages smaller than 500 bytes are discarded because they give troubles for language identification.

Several approaches for pairing pages as translations are considered, such as URL matching and structure matching. Finally BITS uses content-based matching. One possible approach for it is by the use of cognates (similar words in different languages) but if it is not possible a translation lexicon is required. Before the content checking some other checks are made like length and number of anchors.

The reported results are good (97% precision and 99% recall) though the algorithm is quite resource consuming. In 10 nights over 20 stations BITS managed to obtain a total of 63Mb of parallel texts.

2.1.3. PT Miner

Another system called PTMiner (Parallel Text Miner) described in (3), proposes a technique similar to the first STRAND system. PTMiner uses a search engine to find sites with pages in the target languages. Then it fetches URLs of all the web pages indexed by the search engine for the site. A crawler is used which starts from the URLs collected and obtains all the URLs that can be reached from them. This step is necessary as the search engines not always index all the pages and they have limitations on the number of results from the same site. In the final two steps PTMiner does URL matching to generate candidate pairs and uses external features as page size and character encoding to verify them without taking into account the contents of the pages.

Using this system an English-Chinese corpus was constructed containing 14820 pairs of texts in a week. The corpus has a 90% precision.

2.1.4. RSS News Feed Corpus

Another completely different approach for generating a parallel corpus is presented in (4). It proposes a method for generating a growing parallel corpus which is much easier to implement.

The method described relies on two trends of growing practice in the news delivery over the WWW. The first one is multinational news agencies which publish their articles in various languages, translating the original article (normally English) to other languages of interest. The most productive are the news sites in the domain of information technology.

The second trend is the use of RSS for delivering the news. When articles are delivered by RSS in the target language they normally contain a link to the original article which is in English so there is no need to verify if the pairs of texts are actual translations or not, assuring a 100% precision.

In this project a Japanese-English corpus is generated by subscribing to four different news feeds with a RSS client that delivers the news feeds over email. Then the emails are processed using a standard UNIX tool called procmail. With a short configuration file the Japanese article is downloaded followed by the English. Using this methodology corpus of 329 pairs of articles was created over five weeks. Since 329 articles is not enough for any language processing task the four news sites are crawled obtaining additional 15,000 articles. This corpus is growing by approximately 70 pairs of articles per week.

This method is only applicable to high density languages as its success relies on finding news sites that publish translated stories. The domain of these stories is of great importance for obtaining a diverse corpus.

2.1.5. Comparable Corpora

Some other articles (5), (6) propose the use of comparable instead of parallel texts for similar purposes as in this project. Comparable corpora is easier to obtain and more abundant as there are a lot of articles on the same topic which are not necessarily translations of each other. Once again news articles are used for building the corpus.

2.1.6. Existing Corpora

There are several parallel corpora, containing aligned English and Spanish text, which are available for free. Here all the important features of some of those corpora are described.

➢ *Europal Corpus*

This corpus, which detailed description can be found in (7), is extracted from the European parliament proceedings from 1996 to 2006. It includes 11 European languages which means 10 different aligned corpora between English and another of the rest of the 10 languages. One of these corpora is a Spanish-English corpus containing 1,304,116 aligned sentences with 37,870,751 words in Spanish and 36,429,274 words in English.

➢ *JRC-Acquis*

JRC-Acquis(8) is a multilingual parallel corpus constructed by the European Commission Joint Research Center. It contains the total body of the European Union (EU) law including political objectives, treaties, declarations, resolutions, agreements, etc. in 22 of the 23 official languages of the EU. The texts are aligned at paragraph level for all 231 language pairs. The last version of the corpus contains 34,588,383 words in English and 38,926,161 words in Spanish.

➢ *OPUS – an open source parallel corpus*

"The opus corpus is a growing resource providing various multilingual parallel corpora from different domains"(9). It consists of sentence aligned texts collected from the web by open source products without any manual corrections. Some of the corpora included are:

- EMEA – European Medicines Agency documents with 1,145,631 Spanish-English aligned sentences.
- OpenSubtitles constructed form movie subtitles containing 554,526 Spanish-English sentence pairs.
- Europarl3 – another version of the Europal corpus containing 1,304,273 Spanish-English sentence pairs

2.2. Named Entity Recognition (NER)

The second part of the project, after obtaining the bilingual corpus, is extracting all the names from the text. This task is commonly referred to as Named Entity Recognition and Classification (NERC). The term Named Entity (NE) is widely used in Natural Language Processing. It was first introduced at the Sixth Messages Understanding Conference (10), which was discussing Information Extraction (IE) tasks. Back then people started finding necessary to extract names from text, such as people, locations and organizations. It was considered a separate task in IE. All this and much more information about NERC can be found in (11). This paper is a survey of the research in the field of NERC from 1991 to 2006. The evolution can be followed from early systems using handcrafted rules to most modern ones relying on machine learning techniques. The paper gives a clear idea of the complexity of the task of recognizing NEs. This is a very important part of the project and poor results in this phase would severely affect the final performance of the system. Therefore, given the time constraint for the project, instead of investigating more in NERC techniques the research is turned to already developed systems available for free.

2.2.1. Balie

Balie (http://balie.sourceforge.net/) or baseline information extractor is open source software, implemented in Java. It has various features one of which is named entity recognition. The NER system is explained in (12). The paper does not give a clear idea if the system is language independent. The last version of Balie is from late 2007.

2.2.2. FeeeLing

FreeLing (http://garraf.epsevg.upc.es/freeling/) is an open source library providing a lot of language analysis services one of which is named entity detection. It supports English and Spanish among other languages. It provides a Java API but one of the installation requirements is Linux. The last version of FreeLing is from September 2009.

2.2.3. JBL Named Entity Tagger

This system available at http://l2r.cs.uiuc.edu/~cogcomp/asoftware.php?skey=FLBJNE is a free named entity tagger implemented in Java. It is not a library but a Java program which takes as input plain text files and returns the text with the named entities tagged. The system achieves "state of the art performance on the Named Entity recognition task" according to (13), a paper where the design is explained. JBL Named Entity Tagger was constructed for English texts and there is no information about its performance on Spanish.

2.2.4. AlchemyAPI

AlchemyAPI (http://www.alchemyapi.com/) is another language processing library which offers a named entity recognition function. Apart from a SDK it provides an HTTP interface. There versions of the SDK for many programming languages such as Java, Perl, C, C++, etc. It is dual license software limiting its free usage to a maximum of 30,000 API calls a day. To use a free license a registration is required to obtain a license key. Being commercial software it offers support and competitive performance. Alchemy's NER fuction supports English as well as Spanish.

2.3. Names Alignment

The final part of the project takes as input the named entities from the English text and the ones from the Spanish text and has to produce the pairs of names which will be placed in the final lexicon.

One of the most important works on word alignment for machine translation is the IBM Candidate project. Though it is quite old and not state of the art anymore it is referred to in many of the projects in the field for some of its methods are still used. It is important therefore to briefly describe the principles of the IBM Models. A more detailed explanation can be found in (14).

2.3.1. IBM Model 1

IBM Model 1 is a statistical model that gives the probabilities of different translations of a given sentence based on the translation probability distribution of each word in the sentence. What is interesting about it, is how the so called expectation maximization (EM) algorithm is applied to it. EM receives as input a set of translated sentences and discovers the probability that a foreign world *f* is translated to an English world *e* for all foreign words in the input. The in depth explanation of the model includes statistical equations which are not included here because what is of interest for this project are the basic ideas so a simple explanation is intended.

In order to estimate the translation probability of a foreign word *f*, information about its possible translations is needed. If it is known that out of 100 appearances, *f* has been translated 82 times as e_1, 12 times as e_2 and 6 times as e_3 then the lexical translation probability distribution can be estimated as: $p(e_1|f) = 82\%$, $p(e_2|f) = 12\%$, $p(e_3|f) = 6\%$, where $p(e_1|f)$ is the probability that *f* is translated as e_1.

The problem is that when the input is a set of aligned sentences there is no information about the word alignment in each sentence. With other words it is unknown which word is the translation of *f* as it might be the first word of the sentence or the last, or any of the words in between. If the word alignment was known it would be trivial to determine the lexical translation probability distribution as it was trivial in the example before. On the other hand if the probability distribution was know it would be very easy to determine the alignment for the words in each sentence pair.

This is where the EM algorithm comes into play to solve this problem. The algorithm starts with initializing the translation probabilities uniformly, which means that $p(f|e)$ is the same for every *f* and *e* in the corpus. The next step looks for the most likely word alignment in every sentence. In the first iteration, since all the lexical probabilities are equal, all the alignments are equally probable, but further on the translations with higher probability lead to more probable alignments. This step is called the expectation step.

Next is the maximization step. It consists in giving new translation probabilities, by maximum likelihood estimation, counting the co-occurrences of the translation candidates and weighting the alternatives with their corresponding probabilities. The last two steps are iterated until the algorithm converges.

The pseudo-code for the algorithm as written in (14) can be found on Figure 2.1.

```
Input: set of sentences pairs (e,f)
Output: translation prob. $t(e|f)$
 1: initialize $t(e|f)$ uniformly
 2: while not converged do
 3:   //initialize
 4:   count($e|f$) = 0 for all $e, f$
 5:   total($f$) = 0 for all $f$
 6:   for all sentence pairs (e, f) do
 7:     //compute normalization
 8:     for all words $f$ in f do
 9:       s-total($e$) = 0
10:       for all words $e$ in e do
11:         s-total($e$) += $t(e|f)$
12:       end for
13:     end for
14:     //collect counts
15:     for all words $e$ in e do
16:       for all words $f$ in f do
17:         count($e|f$) += $\frac{t(e|f)}{\text{s-total}(e)}$
18:         total($f$) += $\frac{t(e|f)}{\text{s-total}(e)}$
19:       end for
20:     end for
21:   end for
22:   // estimate probabilities
23:   for all foreign words $f$ do
24:     for all English words $e$ do
25:       $t(e|f) = \frac{\text{count}(e|f)}{\text{total}(f)}$
26:     end for
27:   end for
28: end while
```

Figure 2.1: EM Training Algorithm for IBM Model 1**(14)**

In the pseudo-code **e** denotes a sentence in English **f** a sentence in the foreign language, *t(e|f)* is the lexical probability that the translation of the foreign word *f* is the English word *e*.

IBM Model 1 is a good model to start with but it has many flaws. It does not allow traducing one word as two or vice versa, and does not give any information in terms of word ordering. There are four more models, form Model 2 to 5, which try to fix those deficiencies.

IBM Model 2 includes an explicit model for word alignment, Model 3 allows one word to be translated to various or none, Model 4 adds a relative alignment model and finally Model 5 fixes deficiency. All these models are built on top of the preceding model conserving the general principles.

3. Requirements and Analysis

The main requirement of the project is to construct a translation lexicon for English and Spanish names from the Web. In order for this lexicon to be useful it has to cover diverse domains and it has to be updated regularly with new name translations.

This is broken into three separate parts which are integrated in the end.

3.1. Construction of Bilingual Corpus

The first part is to obtain an Anglo-Spanish corpus from which the name translations can be learned. If the lexicon has to be diverse in topics then the domain of this corpus also needs to be diverse. Another requirement is that the lexicon needs to be updated regularly with new entries.

Since the project is on a very short time limit the best decision would be to utilize one of the constructed corpora described in section 2.1.6. The problem is that most of them are on a particular topic and therefore, offer no diversity at all. Apart from that, those corpora are not updated on a regular basis. This means that if one of them is used, the requirements noted in the previous paragraph would not be completed. Those requirements are not primary, but there could actually be a solution that meets them, without producing severe consequences for the planification.

The rest of the literature research from section 2.1 gives a lot of interesting ideas for constructing a parallel corpus. Building a web crawler is maybe the best solution for a diverse and constantly updated corpus. Nevertheless, it is clear that it is not a trivial task, it is very time consuming and if not constructed correctly it might even not produce the expected results. This is the reason to discard this option and look for another one instead.

Collecting RRS feeds, or crawling a News Web site with translated articles, as in the example from chapter 2.1.4, would produce a feasible corpus and is not as time consuming as a general web crawler. Unfortunately, this approach relies on finding Web sites that publish translated articles from English to Spanish or vice versa, and only a few were found, with not enough content in any of them. What was found, were some news agencies that publish the same stories in both languages, even though the articles were not translations of each other. The perfect example for constantly updated and also diverse in topic materials are news articles. Normally the same news story is told by two different journalists in different languages but including the same facts and therefore, the same places, organizations or person names. This makes a comparable corpus very suitable for the project objectives.

One of the Web sites found is of a news agency called Euronews, and it did not take a lot of thinking to decide that its archive is very good for constructing a comparable corpus. What makes this archive so appealing is that it contains articles for each day since the 7th of October of 2004 and links to those articles are available in pages with URLs like http://www.euronews.net/2004/10/07/ for the English versions and http://es.euronews.net/2004/10/07/ for the Spanish ones. There is a picture attached to each one of the articles which is the same for both versions, therefore it is very easy to match the pairs of comparable articles with 100% precision. On the other hand the structure of the HTML for each page is always the same allowing easy extraction of the links and the picture associated, as well as the title and the text of the article leaving out all the irrelevant information on the page.

The archive is updated daily with news from the previous day which makes possible to update the corpus on a daily basis.

3.2. Named Entity Recognition

This part of the project has its only requirement implicit in its name. As discussed in chapter 2.2, it is not feasible to construct a NER system. Therefore, one of the systems presented there will be used. The aim of this chapter is to decide which one to use, but the truth is that any one of them would complete the requisites, and the only way to decide which would perform better is to test them with the corpus in hand. Unfortunately, evaluating the NER systems is a very time consuming task and therefore, will not be performed.

The system that is chosen is Alchemy, described in 2.2.4. The reasons for choosing it before the other ones are the programming language and environment, and the ease of use: Java, Windows and a fairly simple API and good documentation. Furthermore Alchemy is a dual license system (free and commercial) which normally means that it should be good enough in order for someone to consider paying for it, instead of using a free alternative. For the purpose of this project a free license for Alchemy will be used, which limits the API calls to 30,000 a day which should be more than enough.

3.3. Names Alignment

The final and most important part of the project is constructing the lexicon by aligning the NEs discovered in the previous phase. The input to the phase is similar to a sentence aligned bilingual corpus in which word alignment should be performed.

Most of the literature on word to word alignment deals with parallel and not with comparable corpora. Therefore an assumption is made that every word from the English sentence will be translated in the foreign sentence, which is rarely true in a comparable corpus. First of all, the articles not always include the same named entities. Even though the contents of the articles are normally the same, it is sometimes possible to use different names giving the same information, for example one article might say Obama while the other might use the president of the US. Apart from this, the input to this phase comes from the NER phase which is far from perfect. In a good scenario it would give 80% recall. The problem is that NER is performed in two different languages so the 20%, of the NEs, that are not recognized in Spanish might be from the ones recognized in English.

Apart from the recall precision may also be a problem. Even the best NER systems don't give 100% precision especially when the corpus is not in a single domain. Poor precision in the NER phase provides bad input for the final phase in the form of lots of words which are not named entities that might be confused for translations of a properly recognized named entity.

All these flaws in the input data make it quite inappropriate to use any of the techniques for word to word alignment for parallel corpora. Nevertheless, the EM algorithm for IBM Model 1 will be implemented, but only to compare its results with the results of the actual algorithm that will be used. Since the pseudo-code for EM for IBM Model 1 is available, it will be a quite simple task to implement it.

After deciding what shall not be used to solve the problem of names alignment, it is time to decide what algorithm shall solve it. The input to the algorithm is pairs of sets of NEs, one set in English and one in Spanish. Some of the NEs in these sets will be translations of each other, others will not. The algorithm has to decide which the translation pairs are.

It is clear that the decision cannot be made for the individual pairs since there is not enough information to do it. Nevertheless, the more a pair of an English NE and a Spanish one occurs in different articles the more probable it is that they are translations of each other. This is a very simple idea that is going to be used to align the NEs.

On the other hand, a very important feature of the input is that a lot of the entries in one language will be exactly the same as in the other, because person names are almost always the same in English and Spanish. Even when the named entities are not exactly the same many of them are very similar. Therefore, the first decision criteria will be a similarity measure. To quantify the similarity between two strings, the Jaro Winkler algorithm is going to be used as implemented in Sam's String Metrics (15) open source library.

The exact algorithm for constructing the lexicon will be given in the next chapter which explains the design of the system.

3.4. Integration of the Parts

The project is divided into three separate parts but in the end the three of them have to be integrated into one whole. This means that the output of each part will have to be used as input to the next one. And the output of the final part shall be the lexicon. This has to be taken into account for the design and implementation of the software system.

On the other hand it is important to develop the parts as separate modules, so that it is possible to interchange them. Even though it is not going to be done in this project in a future work it might be interesting to change for example the NER system with another one.

The output of each phase should be persistent. The idea behind this is that if some modification is needed in the final phase there is no need to execute the first two phases again.

4. Design, Implementation and Testing

For the development of the software the three phases of design, implementation and testing were almost mixed into one. The reason for this is that the software requirements were quite simple and since the project had to be developed in a very short time, this was considered the most appropriate thing to do. This way it was easy to start implementing with a simple design and then modify and complete it during implementation and sometimes even after testing. This is very similar to what is done in the Extreme Programming methodology.

The programming language to be used is Java. The programmer is very familiar with it and the third party software is also implemented in Java.

This chapter, once again separated into three parts, describes the final design of each part and some implementation techniques that were used, together with the tests that were executed.

4.1. Construction of the Corpus

The construction of the corpus is divided into two parts. As noted in chapter 3.1 the archive of Euronews consists of pages with URLs like this http://www.euronews.net/2004/10/07/ that contain links to the actual articles for the day 07/11/2004. First the archive is crawled generating URLs for every date until today and saving a list of links for all the articles in the archive. To pair the English articles to the Spanish ones the image associated to each article is used. It is the same image for both versions so the URL of the image is matched. It is easy to extract the links and the URL of the images because the HTML structure of the page is very strict and the crawler knows exactly where to look for them. The list is written to a file so that it can be used as input to the second part. If a link fails to download it is skipped, since either the page is down or the HTML structure of the page is incorrect. No special testing is performed for this part only executing it to check if the list gets generated.

The second part consists in reading the list of articles and downloading each one of them extracting only the title and the text of the article. It is easy to extract only the important information because all the articles have the same HTML structure. Each article is saved in a separate file. In order to make the working data more manageable instead of creating thousands of different files all the articles are included in a single zip file. No compression is used since the articles are not very big and compression would slow down the system considerably. Every article is given a consecutive number starting from 1. For example the English article is saved in a file called 1.en.txt and the Spanish comparable article in 1.es.txt.

The decision of using a zip file to store the articles in a single file instead of more than 50,000 different ones was one of the design decisions taken after testing. The 50 thousand files occupied much more space on disk than their actual content size. It was also a very time consuming task to open the directory containing them or coping them from one directory to another.

If an article fails to download it is skipped because either it is offline or its HTML structure is incorrect. This part of the system is tested by executing it and checking that the articles are downloaded.

The two parts explained before are implemented as two static functions in the EuronewsDownloader class. Since one of the software requirements is that the corpus can be updated both functions work in an incremental way. The list of links is continued from a certain date received as parameter to the function. The articles are downloaded starting from a certain number of article, received as parameter, and the number of the last article downloaded is returned. By saving the last date when the corpus was updated and the number of articles in it, it can be updated incrementally with the new articles in the archive.

4.2. Named Entity Recognition

Alchemy provides and API that takes as input a text string and returns an XML with only the named entities extracted. It performs not only named entity recognition but also classification and even disambiguation for English. Although this information will not be used in the project, it will be saved as it could be used in the future. The process of extracting the named entities form the articles is again composed of two parts. The first one is making the API calls to Alchemy obtaining the XML files and saving them, and the second is reading the XML files extracting only the named entities and writing them to separate files.

Again thousands of new files are produced so they are written to zip files without compression.

Some unit testing is performed for both parts to make sure that they work correctly.

The two parts of this phase are programmed in two separate static functions in the NamedEntityExtractor class. Because of the incremental nature of the software the functions receive parameters which indicate the number of the first and the last article on which to execute.

4.3. Names Alignment

Designing this phase is a little bit more complicated. The goal is to read all the input into the program without losing any information. What needs to be saved is the English NEs, the Spanish ones, the number of times they appear, their translation candidates, the number of co-occurrences of a NE and each of its translation candidates and the NEs that appear in each article. Since the term translation candidate was used it has to be defined. If a NE *e* appears in the English version of a particular article and the NE *s* appears in the Spanish one, then *e* and *s* are said to be translation candidates to each other.

The design consists of 3 classes as shown in Figure 4.1. The most important one is the NamedEntity class which represents a NE. It contains the information about the number of appearances of the NE and also a Hashtable with all its translation candidates, where the key to each entry is the name of the translation candidate. The entries are of the class TranslationCandidatePair. A translation candidate pair consists of two references to NEs, one for the English and one for the Spanish, and the number of times these two co-occur. The last class is Lexicon and contains a Hashtable with the English NEs, where the key is the NE, a Hashtable with the Spanish NEs and an ArrayList that contains an entry for each article containing references to the English and the Spanish NEs in that article.

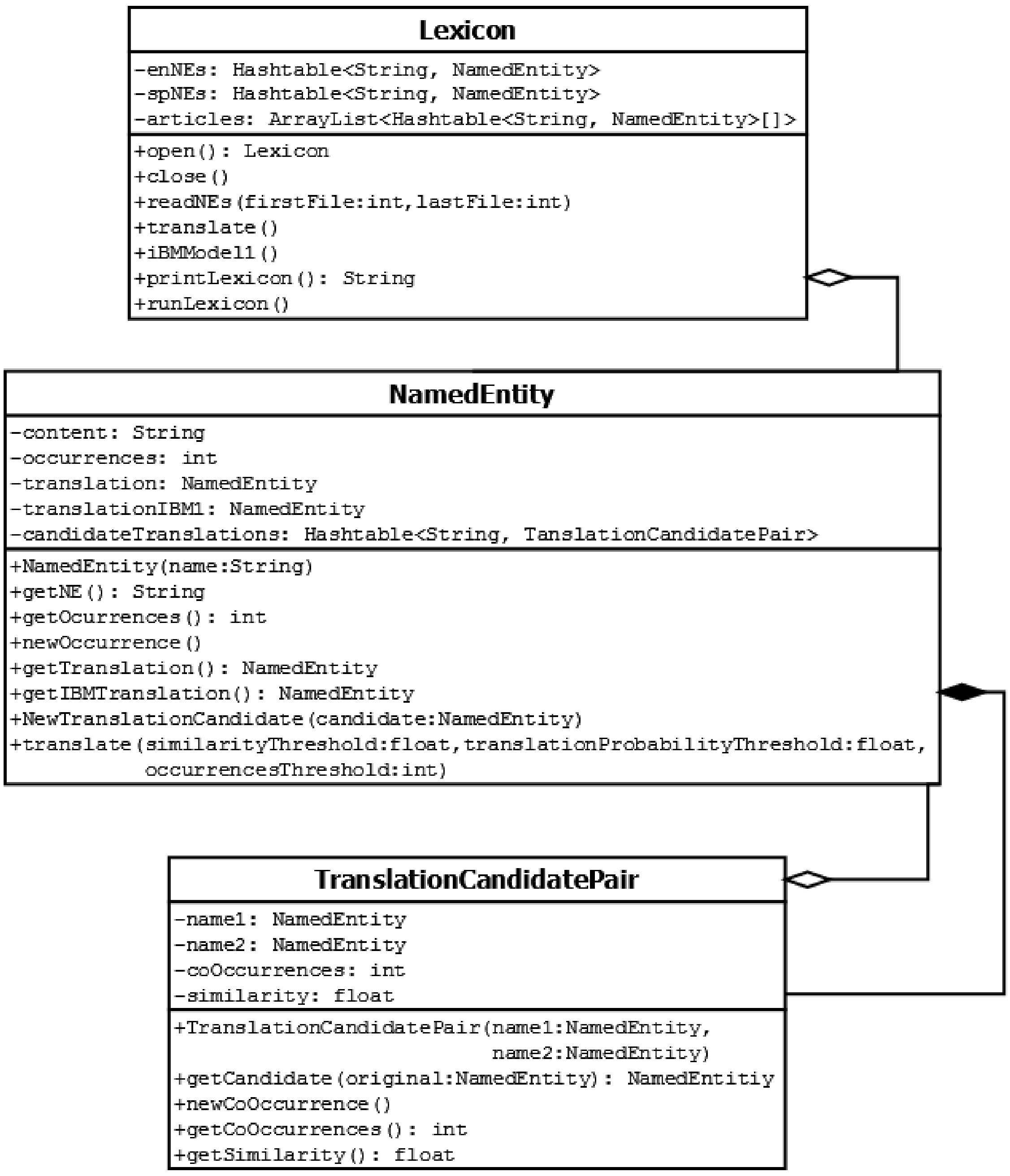
Lexicon
-enNEs: Hashtable<String, NamedEntity>
-spNEs: Hashtable<String, NamedEntity>
-articles: ArrayList<Hashtable<String, NamedEntity>[]>
+open(): Lexicon
+close()
+readNEs(firstFile:int,lastFile:int)
+translate()
+iBMModel1()
+printLexicon(): String
+runLexicon()
NamedEntity
-content: String
-occurrences: int
-translation: NamedEntity
-translationIBM1: NamedEntity
-candidateTranslations: Hashtable<String, TanslationCandidatePair>
+NamedEntity(name:String)
+getNE(): String
+getOcurrences(): int
+newOccurrence()
+getTranslation(): NamedEntity
+getIBMTranslation(): NamedEntity
+NewTranslationCandidate(candidate:NamedEntity)
+translate(similarityThreshold:float,translationProbabilityThreshold:float, occurrencesThreshold:int)
TranslationCandidatePair
-name1: NamedEntity
-name2: NamedEntity
-coOccurrences: int
-similarity: float
+TranslationCandidatePair(name1:NamedEntity, name2:NamedEntity)
+getCandidate(original:NamedEntity): NamedEntitiy
+newCoOccurrence()
+getCoOccurrences(): int
+getSimilarity(): float

Figure 4.1 Class Diagram

The class diagram of Figure 4.1 contains the most important fields and operations. All three classes are serializable which is important for saving the lexicon to a file after it is closed. This design allows maintaining all the information in the memory, giving fast access to it and also it allows accessing it in different ways. Like for example direct access to a particular NamedEntity object using its name as a key, or iterating over all the English NEs, or over all the articles.

Now that was explained how the information is represented it is time to see how the actual translations are made. First of all, all the NEs are read into the memory by the Lexicon.readNEs() operation. This creates all the references and counts all the occurrences of NEs and the co-occurrences with translation candidates. Then the Lexicon.translate() operation is executed, which calls the translate() function of every English NE. This is where the actual decisions about matching NEs are made.

The algorithm, for deciding which the best translation for a given English NE is, considers all translation candidates. Then only the translation candidate that has most co-occurrences is taken into account. If there are two candidates with the same number of co-occurrences the translation cannot be decided. If the best translation candidate appears in more than a certain threshold of the total of articles in which the English NE appears then the translation pair is included in the lexicon. But before this the Spanish NE is also checked to make sure that conditions are completed both ways. This is necessary because there may be an English name that appears in five articles and in each one of their Spanish versions appears the same Spanish name. According to the algorithm this is a valid translation pair but not if it turns out that the Spanish name appears in other fifty articles apart from the five where the English candidate occurs. Furthermore, if the English NE has not appeared in enough articles the data will be considered insufficient to decide a translation.

The similarity between a NE and its translation candidate has more preference than the number of co-occurrences. The first reason for this is because if the two NEs are exactly the same it is almost 100% a valid translation pair. Not always because in some cases in the Spanish version of an article, even though there is a Spanish name, the English version of the name is used. The second reason is because even if certain name only occurs only three times and only one of those times there is a translation candidate that is very similar it is highly probable that those two are a valid translation pair. But in those cases the number of appearances is taken into account too, because even if the similarity measure is very high if the candidate only occurs in 1% of the articles it is most probably not a good translation. This is why there are two thresholds for percent of co-occurrences, one for low similarity candidates and one for high similarity candidates. The high similarity threshold is three times smaller than the normal one.

The thresholds for deciding the translation pairs were calibrated with a lot of testing and investigating the data. Changing the thresholds normally improves the precision but decreases the recall or the other way around. The values selected for the final evaluation of the results are 0.7 for the similarity measure, at least 5 occurrences to look for a translation and at least 51% of co-occurrences for the translation candidates.

4.4. The Main

There is a Main class for the main function of the program, but the important functions of the system are the ones contained in Lexicon class. The main function updates the lexicon printing the new version to a file, and then runs it. Running the lexicon means running a console program which translates on screen keyboard input NEs until ‘exit’ is typed.

For the incremental updates of the lexicon, an *update.log* file is created that saves the date of the last update together with the number of the last article.

Since the lexicon is composed of an enormous number of references the program needs a lot of heap and stack memory for serializing and loading it. To provide that additional memory the program has to be executed with the options *-Xmx256m -Xss64m*.

5. Evaluation of Results

This chapter tries to give a numeric evaluation of the results achieved by the system. There are different measures that can quantify the success of the project and also different parts that can be evaluated. Two of the most commonly used classifications in the area of information retrieval are recall and precision. The recall measures the completeness of the results and the precision measures the exactness.

Given that the project has three different parts it is interesting to evaluate them separately.

5.1. Comparable Corpus

There are several aspects of the corpus that can be evaluated. Maybe the most interesting features are the ones related to its size. The total size of the corpus is 65.8MB and it consists of 27655 pairs of articles with 4,998,462 words in English and 5,145,170 words in Spanish. It is quite big though 7 times smaller than the Europal corpus.

On other bilingual corpora precision might also be an interesting measure since the texts are downloaded and then paired based on different criteria which are not always 100% effective. In the case of the corpus of this project the criteria used for pairing the articles is absolutely precise. There is a small probability of errors since the Euronews archive is not error proof but it is so small that it can be stated that the precision of the corpus is 100%.

It is also important to measure the quality of the corpus for the selected task. In the case of this project a good corpus should contain a lot of NEs, but it is also very important that it contains exactly the same named entities both in the English articles and the Spanish ones. To calculate how suitable the corpus is for the task, the total number of NEs that appear in both languages has to be compared to the number of NEs which co-occur in both the English and the Spanish version of an article. In 30 random, manually expected article pairs, there were 354 total NEs, out of which 159 appeared in both versions of the article. This presents a total of 45%. This means that approximately 45% of the total NEs that appear in the corpus could be included in the lexicon, but the percentage is very low and the problem is that the rest 55% is junk data which could impede the correct functioning of the system.

5.2. Named Entity Extraction

The initial idea was not to evaluate the NER system because it is a third party product not implemented as part of the project. But since the performance of NER has a significant effect on the total performance of the system it was considered important for the final evaluation. The aspects to be evaluated are precision and recall. Because it is a very elaborate and time consuming task to evaluate them only a small number of articles are taken so the results might not be very accurate.

The evaluation procedure consists in examining manually 35 random article pairs in which 265 NEs are discovered in the English articles and 268 in the Spanish ones. Alchemy discovers 240 of which 193 correct in English so the estimated precision is 80.4% and the recall is 72.8%. For Spanish 235 NEs are recognized out of which 190 correctly which gives 80.8% precision and 70.9% recall.

5.3. Lexicon Construction

Two different methods were used for constructing a lexicon so each one of them will be reviewed separately. First the EM algorithm for IBM Model 1 is evaluated, although it not expected to perform well and then the custom algorithm which is intended to give as good results as possible.

5.3.1. EM for IBM Model 1

Given the nature of the data used as input for the algorithm it is expected to perform quite badly especially after seeing that only 45% of the NEs in the corpus co-occur in both English and Spanish articles. Furthermore, the NER system additionally deteriorates the quality of the input.

For evaluating, every pair of NEs that has probability of being a valid translation pair higher than zero is considered valid. It does not sound right that a pair with less than 50% translation probability is correct translation, but an inspection of the results finds out that pairs with very high probability are normally wrong and those that are correct have in many cases less than 50% probability.

The precision of the algorithm is estimated by manually checking the first 100 results out of which 26 are correct and rest 74 incorrect. Therefore, the precision is only 26%.

It is very difficult to calculate the recall, since the total number of named entities is not known. Nevertheless it can be estimated, knowing that the NER system discovered 29,395 unique NEs in English and 27,474 in Spanish. From the English ones only

80.4% are correct which is 23,634 and those are 72.8% of the total number of NEs which gives 32,464 NEs. For Spanish 80.8% are 22,199 correct NEs, representing 70.9% of the total, which is 31310 total NEs.

Now, given that only 45% of the total number of NEs are common for both languages, if C are the common NEs and T the total number of NEs then $T=C/0.45$. And the total number of NEs $T=E+S-C$ where E are the English NEs and S the Spanish ones. Then $C=E+S-T$, substituting T this is $C=E+S-C/0.45$ or $C=0.45(E+S)/1.45=0.45(32{,}464+31310)/1.45=19{,}791$ NEs that can be matched with their translation.

These are rough calculations, because the percentages for precision and recall from chapters 5.1 and 5.2 are far from exact but it is the only information available.

Finally EM for IBM Model 1 gives 9,537 pairs from which approximately 26% are correct, which is 2480 pairs which is only 12.5% of the 19,791 NEs that could be paired. Therefore, the recall of the algorithm is 12.5%.

5.3.2. Custom Algorithm

This algorithm is the one used for the final construction of the translation lexicon so it is very important to evaluate its performance. That is why a big sample of 1,000 entries is inspected for determining the precision. Out of them 939 are correct which gives 93.9% precision. This result is quite good, but having into account that 746 of those entries are spelled exactly the same in English and Spanish it is not such a big achievement. This means that the precision for pairs that are not spelled the same way is 76%, which still is not very bad.

For calculating the recall of the algorithm the calculations made in chapter 5.3.1 are taken into account. Since there are approximately 19,791 valid NEs that can be paired across the English and Spanish versions of the articles, and there are 9760 NEs discovered by the algorithm with 93.9% precision, then there are 9165 valid pairs which is 46.3% recall. This number might seem very low but in fact is not that bad, because maybe more than a half of the NEs only appear in the corpus once or twice which does not give enough information to automatically decide their translation.

As expected, this algorithm performs several times better than IBM Model 1.

6. Future Work

Since the project was severely time constrained, there are a lot of improvements that can be done. In the first place the quality of the software is not very high, both design and implementation could be better. For example multithreading could be done for elaborating the article list and for downloading the articles, since these processes lose a lot of time in waiting the Web response and there is processing work that could be completed meanwhile.

A design improvement could be a better way to represent the lexicon. Serializing and loading a hash table with so many references has proven to be very slow. Maybe the use of a database instead of a hash table could solve that problem and also the problem of needing so much physical memory.

These types of improvements would make the software performance better but would not alter the final results which would be more interesting. The goal is to obtain more proper names in the lexicon and with higher precision. A bigger and better corpus is needed to achieve this, or just more corpora. By developing additional modules the already existing corpora that were discussed in chapter 2.1.6 can be used. It would be also interesting to find more multilingual news agencies since the more texts the better.

In the phase of NER there is not a lot that can be done. Different systems can be tried from the ones described in chapter 2.2 but they would not necessarily be better. It is an interesting option to use various systems, contrasting the results and using only NEs that get recognized by all. This would lower the recall but improve the precision which is more important, since the incorrect results from this stage obstruct the next stage.

Finally for improving the matching of the translations, it would be interesting to consider synonym NEs in English and in Spanish. For example Bush is a synonym for George Bush and George W. Bush. Instead of creating a lexicon of pairs of singe NEs it would be a lexicon of pairs of sets of synonyms. This is a more logical approach, because even though US should be translated EEUU, Estados Unidos is also a valid translation and one that should be used in CLIR. Nevertheless, this opens a whole new problem of creating sets of synonym NEs. On the other hand in this project it is more feasible just to tread the problem as allowing more than one translation for each lexicon entry.

Another improvement to the precision of the final stage can be made by taking into account the types of NEs. Alchemy is actually a NERC (Named Entity Recognition and Classification) system. NEs with different types should never be matched.

7. Conclusions

This chapter will briefly resume the whole work done during the project and will conclude on its achievements.

The most important thing to state here is the primary project objectives were complete. An Anglo-Spanish translation lexicon for proper names was created from

Web resources and it can be updated, with articles included in the Euronews archive every day, by a simple function call.

The whole process of creating the lexicon was divided in three parts. The first was the construction of an Anglo-Spanish corpus. After an extensive investigation and analyzing the different alternatives a comparable corpus was constructed from the archive of a European multilingual news agency. Though the corpus is quite big in size and very diverse in contents it was eventually evaluated as not very good for the task. It contains a lot of NEs but, since the articles included in it are not translations to each other, a big percentage (55%) of the NEs only appear in the English or the Spanish version which makes impossible to include them in the lexicon. And not only that they cannot be included in the lexicon but they make more difficult for the other NEs to be paired.

The next step was extracting the NEs from the corpus and creating pairs of sets of NEs which appear in the same number of article. NER is a very complicated task and constructing a NER system is a whole project on itself. Furthermore the output of this stage is very important for the final results of the project. Those reasons, together with the lack of time, lead to the decision of using third party NER software. The dual license AlchemyAPI is chosen for it is very simple to use. Eventually it was evaluated as not very bad achieving around 80% precision and 70% recall.

The final phase of the project consists in matching pairs of NEs in order to include them in the lexicon. Most of the research on this topic is about word alignment in parallel corpora which is not applicable to this project where, as noted, only 45% of the words can be aligned. Still, an important algorithm which is the base of a lot of works on word alignment is implemented. This algorithm is Expectation Maximization for IBM Model 1. It was later proven, as expected, to give very bad results, only 26% precision and 12.5% recall.

Given that the two languages, English and Spanish, use the Latin alphabet and the matched words are proper names, the key to matching most of the lexicon entries is the similarity between them. Almost 75% of the entries are names spelled exactly the same in both language and another big percentage of them are names spelled very similarly. To match the rest of the entries a count of the pair's occurrences was used. If two NEs co-occur very often then most probably they are translations. This technique has several parameters which had to be tuned in order to obtain acceptable precision and recall. It achieved 76% precision for the 25% of the NEs that are not spelled exactly the same.

The final precision of the lexicon is 94% with 46% recall which is greatly affected by the NER system.

Still there is a lot more to be desired, since the lexicon is not very big and not very accurate and as discussed in the Future Work section it would be interesting to have more than one possible translation for each lexicon entry.

8. References

1. **Resnik, Philip and Smith, Noah A.** *The Web as a parallel corpus.* 2002.

2. **Ma, Xiaoyi and Liberman, Mark Y.** *BITS: A Method for Bilingual Text Search over the Web.* 1999.

3. **Chen, Jiang and Nie, Jian-Yun.** *Parallel Web Text Mining for Cross-Language IR.* 2000.

4. **Fry, John.** *Assembling a parallel corpus from RSS news feeds.* 2005.

5. **Hassan, Ahmed, Fahmy, Haytham and Hassan, Hany.** *Improving Named Entity Translation by Exploiting Comparable and Parallel Corpora.* 2007.

6. **Talvensaari, Tuomas, et al.** *Creating and Exploiting a Comparable Corpus in Cross-Language Information Retrieval.* 2007.

7. **Kohen, Philipp.** *Europarl: A Parallel Corpus for Statistical Machine Translation.* s.l. : MT Summit, 2005.

8. **Steinberger Ralf, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaž Erjavec, Dan Tufiş, Dániel Varga.** *The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages.* s.l. : Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'2006). Genoa, Italy, 24-26 May 2006, 2006.

9. **Tiedemann, Jörg.** *News from OPUS — A Collection of Multilingual Parallel Corpora with Tools and Interfaces.*

10. *Message Understanding Conference - 6: A Brief History.* **Grishman, Ralph and Sundheim, B.** 1996.

11. **Nadeau, David and Sekine, Satoshi.** *A survey of named entity recognition and classification.* 2007.

12. **Nadeau, D., Turney, P. D. and Matwin, S.** *Unsupervised Named-Entity Recognition: Generating Gazetteers and Resolving Ambiguity.* 2006.

13. **Tarinov, Lev and Roth, Dan.** *Design Challenges and Misconceptions in Named Entity Recognition.* s.l. : CoNLL, 2009.

14. **Kohen, Dr. Philipp.** *Statistical Machine Translation.* 2008.

15. **Chapman, Sam.** Sam's String Metrics. [Online] Natural Langugae Processing Group, Departament of Computer Science, University of Sheffield. http://www.dcs.shef.ac.uk/~sam/stringmetrics.html.