UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

BRUNO REZENDE LARANJEIRA

# On the application of Focused Crawling for Statistical Machine Translation Domain Adaptation

Dissertação apresentada como requisito parcial
para a obtenção do grau de
Mestre em Ciência da Computação

Prof$^{a}$. Dra. Viviane Pereira Moreira
Orientadora

Prof$^{a}$. Dra. Aline Villavicencio
Co-orientadora

Porto Alegre, maio de 2015

**CIP – CATALOGAÇÃO NA PUBLICAÇÃO**

Laranjeira, Bruno Rezende

On the application of Focused Crawling for Statistical Machine Translation Domain Adaptation / Bruno Rezende Laranjeira. – Porto Alegre: PPGC da UFRGS, 2015.

68 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2015. Orientadora: Viviane Pereira Moreira; Co-orientadora: Aline Villavicencio.

1. Coleta Focada. 2. Tradução Estatística de Máquina. 3. Adaptação de Domínio. 4. Corpora Comparáveis. I. Moreira, Viviane Pereira. II. Villavicencio, Aline. III. Título.

# AGRADECIMENTOS

Agradeço a Deus.

Aos meus pais e irmãos, por tudo que sempre fizeram por mim, pelo esforço para dar todo suporte para que eu pudesse ser quem sou hoje.

À Camila e à Lara, minha família que tanto amo, por terem tido toda paciência que precisei para realizar este trabalho, sempre dando apoio, amor e carinho.

Aos colegas das salas 215 e 213 do Instituto de Informática, por todas as horas gastas conversando bobagens e ajudando nos vários problemas que surgiram no desenvolvimento deste trabalho.

Aos meus amigos, por todas as conversas, sérias ou não, jogos de futebol, truco e videogame, corridas e momentos bons e ruins.

Às professoras Viviane e Aline e ao Carlos, que me orientaram com toda seriedade e boa vontade necessária.

Enfim, agradeço a todos os que, de uma forma ou de outra, sabem que contribuíram para a minha formação pessoal ou para que este trabalho pudesse ser realizado, pois sei que não teria conquistado isso sozinho.

Também agradeço ao CNPq e ao projeto Cameleon (CAPES-COFECUB 707/11).

*“Science is much more than a body of knowledge. It is a way of thinking. This is central to its success. Science invites us to let the facts in, even when they don’t conform to our preconceptions.”*

— Carl Edward Sagan

**On the application of Focused Crawling for Statistical Machine Translation Domain Adaptation**

## ABSTRACT

Statistical Machine Translation (SMT) is highly dependent on the availability of parallel corpora for training. However, these kinds of resource may be hard to be found, especially when dealing with under-resourced languages or very specific domains, like the dermatology. For working this situation around, one possibility is the use of comparable corpora, which are much more abundant resources. One way of acquiring comparable corpora is to apply Focused Crawling (FC) algorithms. In this work we propose novel approach for FC algorithms, some based on n-grams and other on the expressive power of multiword expressions. We also assess the viability of using FC for performing domain adaptations for generic SMT systems and whether there is a correlation between the quality of the FC algorithms and of the SMT systems that can be built with its collected data. Results indicate that the use of FCs is, indeed, a good way for acquiring comparable corpora for SMT domain adaptation and that there is a correlation between the qualities of both processes.

**Keywords:** Focused Crawling, Statistical Machine Translation, Domain Adaptation, Comparable Corpora.

# RESUMO

O treinamento de sistemas de Tradução de Máquina baseada em Estatística (TME) é bastante dependente da disponibilidade de corpora paralelos. Entretanto, este tipo de recurso costuma ser difícil de ser encontrado, especialmente quando lida com idiomas com poucos recursos ou com tópicos muito específicos, como, por exemplo, dermatologia. Para contornar esta situação, uma possibilidade é utilizar corpora comparáveis, que são recursos muito mais abundantes. Um modo de adquirir corpora comparáveis é a aplicação de algoritmos de Coleta Focada (CF). Neste trabalho, são propostas novas abordagens para CF, algumas baseadas em n-gramas e outras no poder expressivo das expressões multipalavra. Também são avaliadas a viabilidade do uso de CF para realização de adaptação de domínio para sistemas genéricos de TME e se há alguma correlação entre a qualidade dos algoritmos de CF e dos sistemas de TME que podem ser construídos a partir dos respectivos dados coletados. Os resultados indicam que algoritmos de CF podem ser bons meios para adquirir corpora comparáveis para realizar adaptação de domínio para TME e que há uma correlação entre a qualidade dos dois processos.

**Palavras-chave:** Coleta Focada, Tradução Estatística de Máquina, Adaptação de Domínio, Corpora Comparáveis.

# LIST OF ABBREVIATIONS AND ACRONYMS

| | |
|---|---|
| API | Application Programming Interface |
| AUC | Area Under the Curve |
| BFS | Best First Search |
| BLEU | Bilingual Evaluation Understudy |
| CLIR | Cross-Language Information Retrieval |
| FC | Focused Crawling |
| HMM | Hidden Markov Model |
| HR | Harvest Rate |
| HTML | HyperText Markup Language |
| IDF | Inverse Document Frequency |
| IR | Information Retrieval |
| LM | Language Model |
| MERT | Minimum Error Rate Training |
| MT | Machine Translation |
| MWE | MultiWord Expression |
| MWEBFS | MultiWord Expression based Best First Search |
| MWESS | MultiWord Expression based Shark Search |
| NLP | Natural Language Processing |
| NLTK | Natural Language ToolKit |
| PMI | Pointwise Mutual Information |
| SE | Search Engine |
| SMT | Statistical Machine Translation |
| SS | Shark Search |
| TF-IDF | Term Frequency-Inverse Document Frequency |
| TF | Term Frequency |
| UC | Universal Crawler |

| | |
|---|---|
| URL | Uniform Resource Locator |
| WBST | WordNet Based Synonymy Test |
| WCCM | Word-Category Co-occurrence Matrix |
| WSD | Word Sense Disambiguation |

# LIST OF FIGURES

# LIST OF TABLES

# SUMÁRIO

# 1 INTRODUCTION

## 1.1 Motivation

Machine translation (MT) is the process of automatically reading an input text, written in a given source language and, also automatically, translating it into another language (*i.e.*, the target language). MT may be useful for many purposes, from students learning a new language to tourists visiting another country. Translation can be done mainly in two ways: *(i)* by following a set of rules manually defined by a specialist or *(ii)* by learning translation patterns from translation examples. The later, which we call *statistical machine translation* (SMT) is the one we are interested in this work.

Ideally, an SMT system is trained using large quantities of texts written in both source and target languages, known as *parallel corpora*. However, parallel corpora are scarce resources. Most texts available are from religious (*e.g.*, the Bible) or political (Europarl, Hansard) sources, that were manually translated from the original language to many others. This scarcity is more severe for specific domains, like, *e.g.*, dermatology, or under-resourced languages, like Estonian, Icelandic, or Romanian (SKADIŅA et al., 2010).

It is very important for an SMT system to be trained on with data from the same domain as the texts that it will translate. A system trained on generic texts (a newspaper collection) will probably fail to translate domain-specific terms. For example, considering a computer science domain, a generic SMT system that translates from English to Portuguese is likely to mistranslate the word *"shell"* as *"concha"*, when it actually refers to the command line interface. This problem is known as SMT domain adaptation.

An alternative to the parallel corpora may be the use of *comparable corpora*. Comparable corpora are composed of texts that are not exact translations of each other, but that are related to the same topic of interest. Their major advantage is that they are much more widely available and easy to find. The main drawback is that equivalent phrases or sentences must be mined from them (and are not already given as in the case of parallel corpora) and this is a hard task.

Comparable corpora can be acquired in a number of ways. Books related to a given subject or news texts reporting on the same facts, for example, are good sources. In this work, we use *web crawling* as a means to collect them. This is the main mechanism used to traverse a portion of the web graph [1]. Starting from a set of URLs received as input, known as the *seed pages*, the hyperlinks contained in these pages are followed for the neighbor pages to be visited. In order to build a document index, a Search Engine (SE), like Google or Yahoo!, uses a web crawler and stores whatever is convenient from the visited pages, *e.g.*, texts, images, videos and audio files. The hyperlinks found while

[1] *I.e.*, the graph composed considering web pages as nodes and their hyperlinks as edges

parsing the pages are stored in a queue, for the crawler to fetch the pages they point to. Doing this recursively, until some stopping criteria is reached, allows traversing a larger portion of the web graph. Stopping criteria may be the number of collected pages or a time limit for the crawl.

When the crawler is being used to gather texts only from a specific topic, it is not necessary that it visits pages that do not refer to this domain. Hence, it is good for these crawlers to apply some technique that tries to direct the crawling process to visit more pages about that subject, reducing the effort and maximizing the usefulness of the collected material. Crawlers that apply algorithms of this kind are called *Focused Crawlers* (FC).

## 1.2 Hypotheses

The first hypothesis we propose to investigate is that a corpus collected by an FC algorithm can be useful for tasks such as domain-specific SMT. We assess such usefulness by crawling texts on the same topic domain in two distinct languages and then using them to aid domain-specific SMT systems, considering the quality of the SMT system generated translations to be a good indicative of this usefulness.

The second hypothesis is that there is a correlation between the quality of the FC and of the SMT systems that can be constructed by using its collected data. The confirmation of this correlation would indicate that a good FC may be used to build a good SMT system. We evaluate that by comparing one of the classic metrics of FC with some metrics usually used for assessing SMT. We aim to analyze their correlation to observe whether a good evaluation on the FC algorithm, using the usual metrics, does indicate that the resulting corpus is useful. Analyzing the linguistic resources by using them to train a domain-specific SMT system is an *extrinsic* measure, but it is a good indicative of how much the translation system could learn about terms and expressions that are characteristic of the domain of interest and therefore, in a certain way, how the crawling process was able to keep focused to the desired topic.

## 1.3 Contribution

The main contributions presented by this work are:

- Analysis of the correlation between the quality of the FCs and of the quality of the SMT systems that can be built using its collected data.

- Proposal of new FC strategies, based on word n-grams and the expressive power of multiword expressions (MWEs), *i.e.*, group of words whose meaning transcend the combination of the meanings of the component words. We describe MWEs in details in Section 2.2.4.

- Extensive experiments on real web data in which we ran 24 crawler settings.

- Development of a generic, easily extendable FC tool and acquisition of gold-standard data for domain-specific MT for software engineering and dermatology (all available from `http://inf.ufrgs.br/~brlaranjeira/crawler/getcrawler.html`). These domains were chosen because of the availability of humans specialists for translating sentences related to them.

## 1.4 Organization

The remainder of this work is structured as follows:

Chapter 2 contains the basis for the understanding of the subsequent chapters.

In Chapter 3, some relevant related work is presented and discussed. We also point out similarities and differences to our work.

Chapters 4 explains the methodology, detailing both the crawling stage used for acquiring the linguistic resources and the stage where they were used to aid the domain-specific SMT systems.

The experiments and results obtained are presented in Chapter 5.

Finally, Chapter 6 summarizes the contributions and also suggests directions for future work.

# 2 THEORETICAL FOUNDATIONS

This Chapter presents the theoretical foundation necessary for the understanding of the remainder of the work. The following subsections introduce concepts related to information retrieval and natural language processing.

## 2.1 Information Retrieval

Information Retrieval (IR) is the field that aims at the construction of Search Engines (SEs), softwares for searching relevant information related to a topic of interest of the user. SEs usually have a wide collection of documents that are collected by the use of *Web Crawlers*. This is the same mechanism we use for acquiring our collections of documents.

### 2.1.1 Web Crawling

Web crawlers, also called “Web Spiders”, “Web Robots”, or “Bots” (BAEZA-YATES; RIBEIRO-NETO, 2011) are programs whose goal is to automatically download web pages, in order to have their content available offline, as a static collection of texts (LIU, 2011). Although they are mostly used by search engines, like Google or Yahoo! for building their indexes, there are many other applications available for them. For example, they may be used to build a local repository of news, by keeping track of a website, like the Folha de São Paulo web portal [2].

A web crawler starts reading a set of Uniform Resource Locators (URLs), from now on referred as the set of *seed URLs* for crawling. The crawler retrieves and parses the contents pointed by every URL in the seed set. After parsing, the crawler extracts and stores what the final application may consider useful, which may be pictures, videos, structural information or, in most cases, the textual content necessary to build a search engine’s index. The hyperlinks found are also extracted and sent to a structure called the *URLs frontier*, or just the frontier. Then, the URLs in the frontier are followed, extracting the content and the links from the pages pointed by them. This process continues recursively, until any criterion established for stopping is reached. This criterion may be, for example, a time limit for the crawler to run or the maximum size of the collection gathered, in number of pages or logical size (LIU, 2011; CHAKRABARTI, 2002; BAEZA-YATES; RIBEIRO-NETO, 2011).

The documents of the web are in constant change. Therefore, sometimes, an index may need to be updated, either by modifying, adding, or removing documents. Considering its extremely large size (LIU, 2011), it is desirable that a crawler only has to traverse a limited portion of the web graph. If the final application only needs documents referring

[2]http://www.folha.uol.com.br/

to a small set of topics, an alternative is to make the crawler try to traverse the web graph in a directed way, searching only for web pages considered related at least to one of these subjects (LIU, 2011).

There is more than one way to classify web crawlers. LIU (2011) divides them in three main types: Universal crawlers, which are the ones intended to collect any kind of page. Hence, they follow the web graph by simply using a breadth-first search strategy, following the links in the same order that they are found. There are FCs and *topical crawlers*, that are very similar. Both of them aim at gathering pages from a specific topic, but they do it in different situations. Following this definition, an FC makes use of a text classifier (*e.g.*, a Bayesian classifier) in order to determine whether a page is relevant or not and, therefore, needs examples of both relevant and non-relevant pages for the training process. Topical crawlers, in turn, are used when such training data is not available. Thus, the topical crawlers are given only a small set of pages (which can be the same set of seed pages) or words to describe the topic. In short, the main difference between them is the presence or the absence of a text classifier in the algorithm.

The classification done by BAEZA-YATES; RIBEIRO-NETO (2011) divides them in crawlers for general and for vertical web search. Crawlers for general web search have the same definition presented by LIU (2011) for universal crawlers. The ones used for vertical search are tools that restrict their crawling by either the web page's country or dominant language or its main topic. Just like the definition of LIU (2011), crawlers that use the later restriction are also called FCs.

The universal crawler definition of CHAKRABARTI (2002) is the same as the one given by LIU (2011). His definition of FCs, in turn, is very similar to the definition of topical crawler used by LIU (2011). He defines an FC as a system that uses a text classifier to rank pages based on the *Radius-one Hypothesis*. The radius-one hypothesis, as detailed in Definition 1, supports most of the heuristics used by FCs. In Definition 1, the word "positive" is used to indicate a relevant page.

**Definition 1** ***The radius one hypothesis**: if page u is positive and u links to v, then the probability that v is positive is higher than the probability that a randomly chosen web page is positive.*

In this work, we base our definition of an FC on the book of LIU (2011). However, instead of just following his definition literally, we consider both FCs and topical crawlers as FCs, although none of the FCs we experimented makes use of a text classifier. We classify crawlers that apply any algorithm, no matter it uses text classifiers or not, for guiding the crawling process towards relevant pages as FCs. Just like LIU (2011), we say that crawlers that do not use any strategy for this task are universal crawlers.

According to LIU (2011), an FC must have an evaluation independent of the end user application. The argument is that the performance of two FCs used to build the index of two different domain-specific news web portals cannot be compared by measuring how good the search engines of the respective portals are, since there are other algorithms applied during the indexing and the searching stages. Hence, metrics based on the similarity of the collected documents with a user-defined query, like the *Average Precision* (AvP) and the *Harvest Rate* (HR), are used as direct metrics to compare the efficiency of two or more FCs. Both metrics are defined in Definitions 2 and 3 and formalized in Equations 2.1 and 2.2, where $C$ is the set of all collected pages, $q$ is the document corresponding to the query, $\gamma$ is a threshold for distinguishing relevant from non-relevant documents, according to their similarity with the query and $sim(c,q)$ represents the similarity between a document $c$ and a query $q$. The standard metric used for estimating the similarity between two documents [3] is the cosine similarity, but it could also be any other similarity metric. In the cosine similarity, both documents are represented as vectors where the values are the frequencies of the words in the document. Then, the cosine of the angle between the two vectors, which will be a number between 0 and 1, is calculated and used to represent their similarity. One advantage of using the cosine similarity, especially for IR related tasks, is that two documents with very different sizes can be considered similar, thus, not being a problem when we compare short queries with documents of any size.

**Definition 2** ***Average Precision*** *is the arithmetic mean of the similarity of all collected documents with a query.*

$$AvP = \frac{\sum_{c \in C} sim(c,q)}{|| C ||} \tag{2.1}$$

**Definition 3** ***Harvest Rate*** *is the ratio between the number of relevant documents collected and the total number of collected documents.*

$$HR = \frac{|| \{c \in C, sim(c,q) > \gamma\} ||}{|| C ||} \tag{2.2}$$

In addition to these types of crawlers, there is a type of crawler designed to crawl the deep web. The deep web, also called the hidden web is the part of the web that is not accessible by simply following links. Several works exist on hidden web crawling by filling and submitting web search forms and then retrieving the obtained results. However, this is out of the scope of this work. Further reading can be found in RAGHAVAN; GARCIA-MOLINA (2001)

There are some issues common to any kind of crawler, like the guidelines that one must follow. The Robots Exclusion Protocol (JHA et al., 2014), for example, is used by the website developers to restrict the crawler's access. The protocol defines that there must be a file named *robots.txt* in the root of the server and it must specify what paths of the site are not to be followed. This can also be achieved by assigning the value *nofollow* to the *rel* attribute in the HTML anchor tag.

It is also considered impolite if a crawler sends too many requests to a server in a short time interval. The server can easily recognize that those requests come from an automatic (and perhaps malicious) script and, thus, deny them all or respond with garbage information. Hence, the crawler must apply some policy to not overload the servers, either by rearranging the frontier or simply by waiting some time before making consecutive requests to a same server.

Some websites can have *spider traps*. Spider traps are some parts of a website that can, whether on purpose or not, keep the crawler in an infinite loop, in most cases by following dynamically generated links. To avoid this, crawlers often define a maximum number of pages to collect from the same host.

A crawler must also be aware that there are many ways to represent the same URL. As explained by LEE; KIM; HONG (2005), there is a set of five rules that allows these variations. To explain them, it is necessary to know that a URL has the following format: `protocol://host/path:port/#fragment`. For instance, the

[3]In IR models, queries are represented as short documents.

URL `http://localhost/phpmyadmin:8080/#main` is the resulting composition of the *http* protocol, the *localhost* host, the *phpmyadmin* path, the 8080 port and the *main* fragment. The rules for normalizing any URL (*i.e.*, converting any alternative form of a given URL into a unique, canonical form) are: (*i*) converting the characters from the host to lowercase, (*ii*) removing the port component when it is equal to 80, the default value, (*iii*) adding a '/' character to empty paths, (*iv*) decoding escape characters, like a whitespace represented by '%20', and (*v*) removing the fragment. The authors also present three more rules for a more efficient normalization process, but in some cases they may consider two distinct URLs as the same.

### 2.1.2 Cross-Language Information Retrieval

In an IR system or SE there are many ways to improve the query results, in order to make them more likely to show documents that the user finds relevant. One of these ways is to apply relevance feedback algorithms, where the user needs to, somehow, inform the SE what links he considers relevant (MANNING; RAGHAVAN; SCHÜTZE, 2008). The feedback can be obtained by asking the user to check whether a returned document is relevant or not. There is also the possibility of performing a click analysis, which does not require an active collaboration from the user, observing which link the user chose to click first, for how much time he left the resulting tab open and any kind of relevant information. A simple approach is to use the words contained in the documents considered relevant for expanding the user query.

Another strategy for query expansion is to make use of Cross-Language Information Retrieval (CLIR) algorithms. CLIR is an IR application that can benefit from MT, because it deals with multilinguality between the queries and the collection of documents and it. Its goal is to make possible for an IR system to return documents in multiple languages when a user submits a query in another language. Although its execution presents several difficulties, as we explain below, it has the clear advantage of magnifying the collection available for the IR system to seek for relevant documents and, therefore, making its result more likely to bring relevant documents.

GROSSMAN; FRIEDER (2004) mention some questions that should be taken into account when modelling a system that supports CLIR. The first one is about what the system should translate. For this, three options are available: to translate the queries, to translate all the documents or to translate both queries and documents. Simply translating the queries submitted by the user to the language of the collection is the one that represents less computational efforts, since SE collections are usually extremely large. However, it is also the less precise, since the MT system will have access to limited context to perform the translation.

Considering the nature of the queries submitted by users of SEs, it is quite hard to apply MT algorithms that analyze the syntactic structure of the sentence, because, in most cases, they are short and contain only the keywords that describe the topic of interest. Hence, it is only viable to translate some of their words when they are contained in a bilingual dictionary, mapping the concept from the source (*i.e.*, the language used by the user) to the target language (the one in which the documents are written).

If the strategy involves document translation, natural language MT algorithms are applicable, because, unlike queries, documents tend to have syntactically complex sentences. Although more complex MT algorithms tend to generate better translations, they are also very expensive methods, especially when we are dealing with large collections.

Also, the strategy of translating words [4] can be adapted. Instead of performing translations on a single word basis, a system may look for phrases in a cross-language dictionary in order to replace them by one of their equivalents in the other language. Although phrase based translations are shown to obtain better results, it is also really harder to find one in a bilingual dictionary, if compared to the probability of finding simple words, because there is obviously a much larger number of possible phrases for the dictionary to cover.

In a bilingual dictionary, one word or phrase can map to many others in the target language. The question is, then, which of them should be chosen. Just like MT, considering the nature of the queries, it is hard to perform *Word-Sense Disambiguation* (WSD) involving sentence syntax. Although, for this task, it is possible to analyze the words that surround the ambiguous word and compare them to every possible translation to determine which translation is closest to the most appropriate sense of the word in the query. Another alternative is to set weights to the translated words, according to the number of possible translations that they have. If a word has only one or two translations suggested in the dictionary, it is more likely that it is a specific term and, therefore, should receive a higher weight than the ones that seem to be less specific, with many dictionary translations.

If CLIR applies bad MT or WSD algorithms, the quality of the IR system is very likely to decrease. Consider, for example, if a user types the Portuguese word "artilheiro", in a sports related query. In this context, it is quite clear that the user wanted to refer to a soccer player who scores many goals. However, if the SE is not aware of the sports context, it can translate the Portuguese "artilheiro" into the English "gunner", misleading the query to search for military related documents and, therefore, yielding poor results.

## 2.2 Natural Language Processing

In this Section, we briefly detail some of the basic ideas of NLP that are used during the rest of the work, such as the concepts of multilingual corpora, language models (LM), and MT.

### 2.2.1 Types of Multilingual Corpora

MCENERY; WILSON (2001) define a corpus as a collection of texts that satisfies some criteria. The collection should be representative and serve as a standard reference. Also, it must have a finite size and be readable by a machine.

To simplify, we refer to the concept of JURAFSKY; MARTIN (2008), which says that a corpus is an "online collection of text and speech". In this work, we only work with written text collections.

Regarding the languages of the texts that compose the corpora, they can be divided in three main classes: (*i*) monolingual, when all of its documents are written in a single language, (*ii*) bilingual if it contains texts in two languages, and (*iii*) multilingual, when it is composed by documents written in at least three distinct languages. For the sake of simplicity, we are going to group the last two into the multilingual type.

In this work, we are mainly interested in extracting resources from multilingual corpora. Therefore, it is of a great importance that their typology is well defined. AIJMER; ALTENBERG; JOHANSSON (1996) define a multilingual corpus whose documents are exact translations of each other as a *translation corpus*. For the same authors, a *parallel*

[4]As we will see in Section 2.2.3, it is also part of more complex MT algorithms.

*corpus* has texts about the same topic and documents with similar quantities and dates.

MCENERY; XIAO (2007), also define a *comparable corpus* as a set of documents in more than one language, related to a common subject and collected in similar ways.

In addition, using the definition of MCENERY; XIAO (2007), a parallel corpus can be defined as unidirectional or bidirectional. A unidirectional corpus is one where all documents were originally written in one language and then were translated to the other. On the other hand, a bidirectional corpus must have documents originally written in both languages. If the corpus is multilingual, it may also be called multidirectional, by simply following the same criteria.

In this work, we use the definition of MCENERY; XIAO (2007), where a comparable corpus has documents in more than one language, which refer to the same topic and were collected similarly. A parallel corpus must be composed of documents that are exact translations of each other. There also exists monolingual comparable and parallel corpora, although their analyses is not in the scope of interest this work. Monolingual comparable corpora contains texts in only one language related to a common topic, while parallel corpora contain a set of documents whose sentences represent the same meaning[5].

### 2.2.2 Language Models

Basically, an LM is a probability based model for generating sentences in a given language. There are several applications for them. For example, one can use an LM on a text editor, to calculate the probability of a given word to be the next word in a text and decide whether to present it as an autocomplete suggestion. Speech and manuscript recognition and MT systems, as we will see in Section 2.2.3, can make use of LM too (JURAFSKY; MARTIN, 2008).

The basic principle of an LM is the counting of words in a corpus for calculating their probabilities. Knowing how many words [6] a corpus contains and how many times one of them occurs, it is quite simple to calculate its occurrence probability. Instead of counting only single words, we also count sequences of words, or *n-grams*. By counting n-grams, besides of knowing the probability of a given n-gram occurring, comparing the probabilities of every n-gram that starts with an already typed sequence enables ranking them, to generate reasonable autocomplete suggestions.

If we want to know the probability of the occurrence of the 6-gram “The cat is in the bag”, given that it is known that it begins with “The cat is in the”, we must calculate the ratio between the probability of the occurrence of the entire 6-gram and the 5-gram. This intuition is formalized in Equation 2.3, which states that the probability of the occurrence of a word after an (n-1)-gram depends on the probabilities of both the n-gram and the (n-1)-gram.

$$P(w_n|w_n^{n-1}) = \frac{P(w_1^n)}{P(w_1^{n-1})} \tag{2.3}$$

However, calculating the probability of long n-grams is very expensive. To reduce this cost, the *Markov assumption* is used. In short, the Markov assumption says that the future events occurring in a process depend only on its present state, not on its past. Applying this idea to the LMs, we can assume that the next word to appear depends only on the last

[5]A set of translations of the same original document, made by different translators, represents a monolingual parallel corpus.

[6]Note that we are not talking about distinct words.

*n* words. In the example, the probability of the last word to be "bag" depends only on a few previous words, say, "is in the".

Although this is a quite simple way to compute the n-gram probabilities, there are some cases that need to be addressed. If an n-gram does not appear in the training corpus, two problems show up. The first is that its probability is zero, whereas it should have some positive probability. The other is that Equation 2.3 would include a division by zero. One solution is to apply *backoff* algorithms, like adding one to the count of every word or n-gram. For more details on such algorithms, please refer to (JURAFSKY; MARTIN, 2008).

Additionally, other features can be used in the LMs. One can use part-of-speech tags (*e.g.*, noun, verb or adjective) to differentiate words in their context. The sentence "Can I can a can" is a good example where the words would not be distinguished, if we did not use part-of-speech information.

### 2.2.3 Machine Translation

MT is the subfield of NLP whose objective is to read an input sentence written in one language, henceforth called the source language, and produce an output sentence in another one, the target language (JURAFSKY; MARTIN, 2008). Even when done manually, translating a text is a hard task, for many reasons. Many of the difficulties arise from the differences in the source and target language structures. For instance, a usual sentence in English or in Portuguese starts with the subject, followed by the verb and the object, if any, while in Japanese the order of a typical sentence is different, with the object preceding the verb. Some languages, like Japanese and Chinese, impose a greater challenge for tokenizers, because they do not use the whitespace character to separate words. German also has some cases where many words are concatenated into just one.

As already explained in Section 2.1.2, one entry in a cross-language dictionary can map to many entries in the target language. For instance, the verb "know" in English can be translated to Portuguese using the words "conhecer" or "saber". Both verbs have different meanings in Portuguese, but have the same English translation. The French translation to the English word "leg" can be "patte" (animal leg), "pied" (chair leg), "jambe" (human leg) or "étape" (journey leg). To make the reverse process, translating the French "patte" to English, has the same problem, because besides "leg", "paw" and "foot" are also possible translation, depending on the sense it expresses in the sentence.[7]

In JURAFSKY; MARTIN (2008), four main models of MT algorithms are presented: (*i*) the transfer model, (*ii*) the interlingua representation model, (*iii*) the direct translation model, and (*iv*) the statistical translation model. In the transfer model, the input sentence is parsed and converted to a structured representation of the source language. This representation is, then, converted to another structured representation, that models the target language. Finally, the output is generated from this target language structure. The interlingua model is similar, but it uses an internal semantic representation, in which a parsed text written in any language must be represented, and from which the output text, no matter what is the target language, must be generated. The later model reduces the number of algorithms needed for an MT system to cover a large number of language pairs. In the transfer model, this number is quadratic, while in the interlingua, it grows linearly.

[7]Moreover, according to JURAFSKY; MARTIN (2008), one of the greatest challenges in translation can be represented by the Sapir-Whorf or linguistic relativity hypothesis. It states that languages affect the way that thoughts are constructed by the speakers. Therefore, it becomes very hard, not to say impossible, to represent some ideas in a language different from the one in which the main idea was originally expressed.

However, it may be much harder to develop a unified semantic representation, in which texts written in any language can be represented.

The rationale of direct translation is to minimize the computational cost. The translation is done just by performing a few steps, like morphological analysis and rearrangement of prepositions. It is usually suitable for language pairs including similar languages, like Portuguese and Spanish or Catalan.

The fourth model, corresponding to the statistical translation model, is the one that is more related to this work. The idea is to exploit parallel corpora and use them to learn how to translate. A parallel corpus must be aligned at the sentence level and the aligned sentences also must be aligned at the word level. Then, a dictionary can be constructed, mapping source words to its translation in the target language, and estimating the corresponding translation probabilities.

Since an SMT algorithm is not based on rules, but on effectiveness estimation, it needs a function to evaluate the quality of the translation, in order to compare to others. This evaluation is done by assessing the fluency of the output sentence (*i.e.*, how similar it is to a human-generated sentence) and its correctness, comparing to the input. The fluency can be estimated using language models, explained in Section 2.2.2. An estimate of the correctness of the translation can be computed by combining the probabilities of each word in the input sentence to be translated to any word in the output.

### 2.2.4 Multiword Expressions

In many NLP applications, dealing with multiword expressions (MWEs) is a great challenge. SAG et al. (2002) define an MWE as an interpretation of a sequence of words that transcends the boundaries between the words. In other words, it is a sequence of words whose meaning cannot be understood by combining the meanings of its component words. A good example is the expression "take it easy". If one uses it, the objective is not to tell anybody to grab an object easily, like the combinations of the meanings of the three words would suggest. Instead, the expression tells someone to act calm. We call this property the *non-compositionality* of an MWE.

An MWE can be of two main types: a lexicalized phrase or an institutionalized phrase. The former is an expression that has at least one non-compositional part. The definition of the later is paradoxical, because they are not necessarily non-compositional. This group is composed by expressions whose frequency of use is much higher than any other expression that could express the same meaning. For instance, "traffic light" and "bus stop" are much more used than "intersection regulator" and "the place where the buses stop". Although both expressions are compositional, they are MWEs for this reason.

Lexicalized phrases can also be subdivided into three types, according to their flexibility. The first, and the less flexible one, is the fixed expressions group. MWEs of this group are immutable, not allowing syntactic variations. The expression "in short", for example, cannot be turned into "in very short". The second is the group of semi-fixed expressions, that also have their words in a fixed order, but at least one of them allows some kind of modification. The expression "to spill the beans" can be used in a sentence in the past changing it to "spilled the beans". The computer scientist "Donald Ervin Knuth" name is frequently referred omitting the middle name, calling him just "Donald Knuth". The third, and more flexible group, is the group of syntactically flexible expressions. They allow more types of variation in their structure. The MWE "pull a rabbit out of a hat" can be changed into "she could have pulled a rabbit out of her hat". Phrasal verbs, like "wake up", which can be turned into "wake me up", also belong to the last group.

There are many ways to treat MWEs in an NLP application. The simplest way is to treat them as *words with spaces*. Considering “ad hoc”, a fixed expression, as a regular dictionary entry is an example of this technique. To identify syntactically flexible MWEs, like the expression “throw under the bus” in the sentence “He would have thrown anybody under the bus, if he needed.” requires a more complex strategy, like parsing the sentence and identifying its syntactical dependencies to find the MWE.

For an application or knowledge base to learn some new MWE, it is possible to analyze a corpus and use information about words co-occurrence frequency, like the dice coefficient or *Pointwise Mutual Information* (PMI). PMI measures how much two words are associated and is formalized in Equation 2.4, where $p(w_2|w_1)$ is the probability of occurrence of a word $w_2$, given that the word $w_1$ has just occurred and $p(w_2)$ is the marginal probability of occurrence of the word $w_2$. The lower the PMI, the less the two words attract themselves. The higher it is, the more they are attracted to each other. Negative PMIs indicate that one word repels the other. The PMI of the 2-gram “Michael Jordan” is likely to be high, because the words probably occur together many times. Also, it is intuitive that, given that the first word is “Michael”, the probability of next one to be “Jordan” or “Jackson” is higher than another random word.

$$pmi(w_1, w_2) = \log \frac{p(w_2|w_1)}{p(w_2)} \tag{2.4}$$

### 2.2.5 Distributinal Word Similarity Metrics

There are several NLP applications that could use information about word synonymy. Text and speech generators can use synonyms to avoid using always the same words to refer to something. Question-Answering applications can understand a much wider variety of questions, by guessing that some unknown word is synonym of another, well-defined one.

The best way to do this would be to exploit a knowledge base, like WordNet, that contains information about word relations, *e.g.*, synonymy, antonymy, hyponymy and hypernymy (FELLBAUM, 1998). However, there are alternatives for cases when such base is not available. It is possible to analyze a corpus and extract some related word pairs, relying only on their distribution.

The principle for most distributional word similarity metrics is analogous to the popular saying “tell me who your friends are and I will tell you who you are”. In the same manner, we may also describe a word by observing which words it co-occurs with. For every word, the idea is to build a vector, which we call the *context vector*, where every component contains its strength of association with another word. The strength of association can be any association metric, like the probability of their co-occurrence, given that the first (*i.e.*, the word described by the vector) occurs, or the PMI of the word pair.

Once the context vector is built, one can determine which words are related. It is important to say that we are not necessarily finding synonyms, but related words, because antonyms, hyponyms and other kinds of related words also may co-occur with the same words. Just like in the LMs, additional features can be used in the context vectors. For example, a vector can only account for words that are syntactically related [8] in a sentence.

There are three main metrics for finding related word pairs and almost every other metric is based on one of them. Two of these metrics, shown in Equations 2.5 and 2.6

[8] *E.g.*, one is the object of the other, that is the verb.

estimate the distance between the words, *i.e.*, how different they are. The other one, detailed in Equation 2.7, instead, expresses how similar they are.

$$Manhattan(w_1,w_2) = \sum_{w \in C(w_1) \bigcup C(w_2)} |S(w,w_1) - S(w,w_2)| \tag{2.5}$$

$$Division(w_1,w_2) = \sum_{w \in C(w_1) \bigcup C(w_2)} |\log \frac{S(w,w_1)}{S(w,w_2)}| \tag{2.6}$$

$$Product(w_1,w_2) = \sum_{w \in C(w_1) \bigcup C(w_2)} \frac{S(w,w_1) \times S(w,w_2)}{scale} \tag{2.7}$$

In all three Equations, $C(w_1)$ and $C(w_2)$ refer to the context vectors corresponding to the words that are being compared and $S(w_1,w_2)$ represents the strength of association, *i.e.*, how much they are associated, according to any metric. Note that in Equations 2.5 and 2.6, differences between the context vector positions increase the function's result, confirming that it is a measure of semantic distance. The function in Equation 2.7, in turn, decreases its output with the differences, showing that it is a measure that expresses semantic relatedness. The cosine similarity metric (mentioned in Section 2.1.1) is based on Equation 2.7, where the scaling factor is the product of the norm of $C(w_1)$ and $C(w_2)$. It is probably the most popular metric in distributional models.

Another commonly used measure that is based on Equation 2.6 is Lin's measure, shown in Equation 2.8, where $pmi(w_1,r,w_2)$ is the PMI between the words $w_1$ and $w_2$ with the syntactic relation $r$ and $T(w)$ is the set of pairs of words and syntactic relations that co-occur with $w$ (LIN, 1998). Although the Equations look quite different, the idea of obtaining the ratio between two strength of association measures remains.

$$Lin(w_1,w_2) = \frac{\sum_{(r,w) \in T(w_1) \bigcap T(w_2)} (pmi(w_1,r,w) + pmi(w_2,r,w))}{\sum_{(r,w') \in T(w_1)} pmi(w_1,r,w') + \sum_{(r,w'') \in T(w_2)} pmi(w_2,r,w'')} \tag{2.8}$$

# 3 RELATED WORK

In this Chapter, we discuss some of the related work. Section 3.1 revises the relevant literature on focused web crawling, Section 3.2 is devoted to SMT and Section 3.3 is about the construction of distributional thesauri. Then, Section 3.4 presents a summary on them, together with the main differences and similarities between them and our work.

## 3.1 Focused Web Crawling

In this Section, we briefly present some of the relevant work on FC. Most of the algorithms designed for this purpose are based on the radius one hypothesis, presented in Definition 1. Section 3.1.1 presents some simple strategies for FC, followed by some more complex ones, in Section 3.1.2. Then, Section 3.1.3 discusses some alternative approaches.

### 3.1.1 Simple Focused Crawlers

The Fish-Search algorithm, presented by DE BRA et al. (1994), is based on a metaphor with a school of fishes to propose a methodology for focused crawling. The fishes are dependent on food and, therefore, always try to swim in the direction of healthy waters. If they stay out of food longer than they are capable of surviving, or they get into polluted waters, they die. Analogously, the algorithm tries to visit pages with high probability of containing some relevant information (fish food). Each page in the frontier represents a fish. If a path leads to many consecutive irrelevant pages or faces many communication problems (polluted waters), the fish dies.

The algorithm is simple and can be tuned with three parameters: (*i*) *width*: the maximum number of URLs that a page can add to the frontier, (*ii*) how many consecutive irrelevant pages can be found before a fish dies, and (*iii*) minimum transfer rate to keep following pages from a server. The only mechanism used to focus the crawling process is that, if a path of the web graph only seems to lead to irrelevant pages, the crawler stops following it. Hence, there is no weighting strategy for adding the URLs to the frontier in a way that the most relevant ones get higher priority.

Based on the Fish Search, the Shark-Search algorithm was presented in HERSOVICI et al. (1998) and has the goal of overcoming some of its flaws. For this, it uses a strategy for assigning a score for any page, by estimating its relevance. To calculate the score of a given page, two factors must be weighted. The first is the inherited score, that is based on the relevance and on the inherited score of the page that points to the scored page (*i.e.*, the pointer page). The other factor, the neighborhood score, weights the text of the anchor of the pointer link and the text around it. The Shark Search algorithm was the first to differentiate links by their place inside the page and their respective neighborhoods.

### 3.1.2 Advanced Focused Crawlers

The terminology *focused crawler* is introduced in CHAKRABARTI; BERG; DOM (1999), although the same idea is already used in earlier works such as the ones presented in Section 3.1.1. It proposes a technique that uses a hierarchical organization of topics, like the tree of directories from Yahoo! (2012).

A set of non-redundant topics (*i.e.*, one topic can not be a generalization of another) must be defined by the user. For every topic from this set, a set of documents must also be specified, in order to describe it. It is essential that a page refers only to one of the topics from the set. Then, those pages are used to train Bayesian classifiers, that will estimate the probability of a page belonging to a topic.

The score of a page is calculated by the probability that it belongs to any of the topics from the set specified by the user. This score is the sum of the individual probabilities that this page belongs to each topic. Each of the individual probabilities of a page $p$ belonging to a topic $t$ is calculated as shown in Equation 3.1, where $t'$ is the topic immediately superior to $t$, $Pr(t|p)$ is the probability of $p$ to belong to $t$ and $Pr(t|p, p \in t')$ is the probability of $p$ to belong to $t$, given that it belongs to $t'$.

$$Pr(t|p) = Pr(t'|p) * Pr(t|p, p \in t') \tag{3.1}$$

This probability is calculated recursively, until the probability of the root of the topic hierarchy is reached. The probability of a page to belong to the root is 1, because all other topics derive from it.

The method also uses distillers to identify important hub pages, which are web pages that link to many important authority pages, which, in turn, are pointed by many important hub pages. Formally, we can calculate the authority and hub scores of a page as shown in Equations 3.2 and 3.3, where $a(v)$ is the authority score of a page $v$, $h(u)$ is the hub score of a page $u$ and $E$ is the set of all links $(u, v)$, directed from $u$ to $v$, of the analyzed portion of the web graph.

$$a(v) = \sum_{(u,v) \in E} h(v) \tag{3.2}$$

$$h(u) = \sum_{(u,v) \in E} a(v) \tag{3.3}$$

Intuitively, we can understand them as circular concepts, since one needs the other to be calculated. These values can be calculated iteratively, until they converge. Once the distiller has identified potential hub pages, they and the pages linked by them can have their priorities raised.

Still, the user must assess some pages obtained during the crawling process and signalize them as relevant or not. This judgment is used to improve the classifier's training on deciding whether a document is considered relevant.

Good results can be achieved by using this algorithm. However, a large amount of human effort is necessary, making its usage inconvenient or, in some cases, even impractical. Hence, this work exposes the need for developing an algorithm that depends less on manual efforts.

The use of context graphs for finding documents that are not relevant to the topic but may be useful to find relevant ones is also an alternative for FC. When a specialist is browsing to seek for pages referring to a topic of interest, he may find convenient to visit some pages that are not about this subject, but will probably lead to them. For example, it may not be a fair crawling policy to assign a low priority to a university page at the end of the frontier, if the crawler's objective is to seek genetics related documents. Based on this and on the observation that assigning a high priority to those pages is a hard challenge to FCs, the main idea is to identify what kinds of pages use to point to the relevant, target, ones.

For doing this, the context graph around the targets is built using a SE with the capability of retrieving backlinks [9]. Given a target page, the backlinks of its URL are retrieved by the SE. The backlinks of these pages are also retrieved, recursively, until a desired link distance is reached. All pages are labeled according to their distance to a target page, that receives the label 0.

Then a set of Bayesian classifiers is built. Each classifier must decide whether an input document is from a given label (positive classification) or not (negative) and compute the likelihood of the classification.

During the crawling process, a set of frontiers is maintained, one for every label. When a new document is found, it is downloaded, classified into the label whose classifier considered it as positive with the greater likelihood (if all likelihoods are below a threshold, it receives the "*others*" label), and inserted in the corresponding queue. The documents in the frontiers are ordered by the likelihood computed by the corresponding classifier. When the crawler needs to download more pages, it chooses links in the first page of the frontier with the lowest label. The classifiers can also be retrained during the crawling. When a relevant page is identified, the backlinks retrieval procedure is repeated and the classification process can be updated.

Following the same idea, the work of LIU; JANSSEN; MILIOS (2006) tries to learn with specialist users, monitoring some of their browsing sessions. While browsing, specialists signal when they find something relevant. This avoids the need for observing the whole context around the target pages. The method consists of building a Hidden Markov Model (HMM) with the information obtained during the observation sessions. First, all visited pages are used as input to train an XMeans clusterer (PELLEG; MOORE et al., 2000). Then, the graph composed by all visited pages is built and the distance, in number of links, of a node to a relevant one is assigned as its state and, finally, the HMM is built. The hidden states are represented by the states of the graph, the observations are represented by the clusters and the initial probabilities are equally distributed.

During the crawl, when a new page is collected, it is primarily classified into one of the clusters. The position of this page in the frontier is based on the probability that it belongs to a low-numbered hidden state. For instance, if the probabilities of page $u$ belonging to states 0 and 1 are, respectively, 0.2 and 0.3, and the probabilities calculated for page $v$ are 0.2 and 0.4, then $v$ will receive a higher priority in the frontier than $u$. If the probabilities of both being at state 1 were also equal, the next criterion would be the probability of them being at state 2 and so on. Although this is an efficient method, it requires that a group of specialist users browses relevant pages so that the crawler can learn some interesting patterns, which can, in some cases, be prohibitive.

Another possibility is the use of automata that adapt their actions according to the response of the environment, seeking to minimize the negative reactions and to maximize the positive ones, as proposed by AKBARI TORKESTANI (2012). During the crawl, a given document $i$ is represented by an automaton $A_i$, that has a set $u(i, j)$ of links pointing

[9]Page $u$ is a backlink of page $v$ if $u$ has a link pointing to $v$.

to page $j$. The set of actions that $A_i$ is able to execute is $\alpha_i = \{\alpha_i^j \mid \forall \mathrm{u}(i,j)\}$. $\alpha_i$ is variable because it does not consider links that were already visited.

There is a stack with pages to be visited that is initially filled with the seed URLs. A given page $i$, which is at the top of the stack, is visited, its similarity with the query is computed, the set of links $u(i,j)$ is extracted and a new automaton $A_i$ is assigned to it. Initially, all actions that the automaton can execute have their probability equally distributed. Then, a page $j$ must be pseudorandomly chosen, respecting these probabilities.

If the similarity between $j$ and the query is higher than a variable threshold, (that belongs to $A_i$ and has, initially, the value zero), the value of the similarity is assigned to this threshold and the action $\alpha_i^j$ is rewarded, by having its probability augmented. For the page $j$ to be added to the top of the stack, its similarity must be higher than another, pre-established, threshold. Otherwise, $A_i$ keeps executing its possible actions, until it does not have any action available ($\| \alpha_i \| = 0$) and, consequently, $i$ is removed from the stack. When the stack gets empty, a new crawling iteration is started, considering the probability values obtained at the end of the previous one.

### 3.1.3 Alternative Approaches

The use of focused web crawlers in the acquisition of comparable corpora in order to train domain-specific MT systems is proposed in the work of TALVENSAARI et al. (2008). Their algorithm is divided in two main stages: the first was executed by an FC and the second was responsible for finding parallel subcorpora in the acquired comparable corpora.

The crawling is also divided in two steps. The first step consists in manually submitting some queries with keywords from the domain of interest to a SE. The most frequent words contained in the results returned by the search engine are used to build fifty more queries, whose results are scored according to their frequency and ranking in the search result. The second stage is crawling using as seeds the URLs with the highest scores in the hosts whose pages had the largest sum of scores. The score given to each page in the frontier is measured by the ratio of relevant words [10] in the anchor text where the link was found, in the pointer page and in the set of pages belonging to the same host as the pointer page.

The alignment procedure was started by building a query for each of the source language paragraphs. Then, they were translated to the target language using the Utaclir query translator (KESKUSTALO; HEDLUND; AIRIO, 2002), a dictionary based query translator, and FITE-TRT (PIRKOLA et al., 2006), a word translator based on rules learned from a bilingual word list. Then, the translated queries were used as input in an IR tool, that gave a score for every target language paragraph, according to its similarity with the translated query. Documents whose similarity exceeded a given threshold were considered parallel. They conducted experiments using the genomics domain in English, German, and Spanish. Unlike our work, their experiments were limited to CLIR and to single word translations. For these tasks, the corpus obtained at the end of the process showed significant improvements when compared to a larger but generic corpus.

An alternative methodology to FCs for acquiring comparable corpora is proposed in the work of GRANADA et al. (2012). The method uses domain-specific multilingual ontologies and a SE. The algorithm first extracts the labels of the ontology concepts. Then, label pairs are formed by generating all possible combinations of two labels from the

[10]*I.e.*, if it is present in a query, built with the same words used to build the previous fifty queries.

same language. All pairs are concatenated with a fixed term, that describes the domain of interest, creating the set of queries submitted to a SE. The documents pointed by the 10 best ranked retrieved URLs are then collected. Documents that impose some difficulties for reading, like pdfs, images, or other types of encoded files, are discarded. The collected documents are cleaned, keeping only letters, numbers or punctuation marks. Dates, URLs, e-mail addresses, menus and footnotes were also removed. Finally, the texts were linguistically processed: words were lemmatized, part-of-speech tagged and parsed.

## 3.2 Statistical Machine Translation

This Section presents some of the relevant work on SMT related to this thesis. The first and second sections are devoted to SMT evaluation and domain-adaptation, respectively. In the third section, works on the construction of distributional thesauri are discussed.

### 3.2.1 Evaluation

One of the most widely used metrics for evaluating MT systems is presented in the work of PAPINENI et al. (2002). The Bilingual Evaluation Understudy (BLEU) is a metric that assesses the quality of the sentences generated by an MT system by comparing it to one (or a set of) gold-standard translations and assigning it a score between 0 (poor translation) and 1 (perfect translation).

BLEU evaluates the proportion of n-grams, whose order usually ranges from 1 to 4, that are common to the automatic and to the gold-standard translations, combining them at logarithmic scale, because when evaluating high order n-grams, this proportion tends to decrease drastically. Also, it applies a penalty for very short sentences and does not consider words or n-grams when they are repeated more times than they happen in the gold-standard. The former must be done because very short sentences may have all its n-grams in the reference translation, but this does not mean that it is a good translation. The later is needed for penalizing systems that repeat some words or passages more times than necessary, even if they are correct.

Besides BLEU, there are many automatic metrics for assessing the quality of MT, by comparing a set of automatic translations to one or more gold-standards. These metrics, however, do not tend to agree with each other, when comparing different translations, mainly because their goals are different. Some metrics compare sentences at lexical or phrasal level, others make a syntactic comparison of their structure, while some can use knowledge bases like WordNet to identify when a translation uses a synonym of a word or phrase used in the gold-standard.

The work of GIMÉNEZ; AMIGÓ (2006) aims at providing a framework for a proper use of a set of MT quality assessing metrics, without the need of user-defined parameters for weighting them. The framework has three main components: $QUEEN$, $KING$ and $JACK$. $QUEEN$ is the component whose objective is to determine the probability of an automatic translation to be considered a gold-standard (*i.e.*, to be a good translation). The $KING$ module evaluates the reliability of a set of metrics, by computing the probability that the $QUEEN$ value for a gold-standard is higher than for an automatic translation. The last component is $JACK$, whose goal is to evaluate how reliable a test set is, computing the probability that a pair of automatic translations with $QUEEN > 0$, to be closer to a reference than to each other.

The framework outputs a final score for each MT system, corresponding to the $QUEEN$ value for them using a set of metrics. To define this set of metrics, the $KING$

score for every metric is computed and, one by one, they are added to the metric that has been assigned to the higher value, while the $KING$ value for the set keeps increasing.

In the experiments presented by those authors, there was no agreement among the algorithms as to which MT system presented was the best. The evaluation generated by the framework was coherent to what was expected by previously knowing the MT systems. They did not present, however, an evaluation comparing it to human evaluation.

In IRVINE et al. (2013), two MT evaluation methods to assess the effects of using a generic SMT system to translate domain-specific sentences are proposed: WADE (Word Alignment Driven Evaluation) and TETRA (Table Enhancement for Translation Analysis). WADE observes translations at the lexical level, classifying each error in a taxonomy called the $S^4$ Taxonomy. This taxonomy defines that any lexical choice error made by an SMT system can be assigned to exactly one of those errors: (*i*) SEEN: when the source word is not in the phrase table, (*ii*) SENSE: when the source word is in the phrase table, but translated as another sense, (*iii*) SCORE: when the correct translation pair is in the phrase table, but another word is chosen or (*iv*) SEARCH: when the error occurred due to pruning in the beam search, but this error was not considered in WADE, because it does not represent a significant source of errors, if using a large enough beam size.

TETRA evaluates what could happen to the performance of the SMT systems when in-domain parallel data is added for training them. Based on the errors found in WADE, four new SMT systems were built. Two of them meant to correct SEEN and SENSE errors, by adding word pairs with the same source word and two others for fixing the SCORE errors: one using values of a generic phrase table and another combining these values with in-domain data.

The analyses indicated that most of the errors are of the SEEN type and that merely adding examples with the same source word to the training data is not enough to significantly improve the SMT quality. This is because many of these errors just become SENSE or SCORE errors. Also, most of the errors occurred with nouns.

### 3.2.2 Domain Adaptation

Domain-specific terminology, with a very particular usage, poses well-known problems for SMT systems trained on generic corpora. For example, a generic domain SMT system may fail to identify that "*shell*" in Computer Science jargon refers to a command-line interface rather than to the carapace of an animal. Therefore it is not surprising if an SMT system performs poorly in new domains.

LÄUBLI et al. (2013) approach the problem of domain adaptation for SMT by using domain-specific parallel data. The proposed idea is to combine both in-domain and out-of-domain data, since finding large enough in-domain data is hard. Instead of using a discrete classification of whether a corpus is in-domain, a continuous value is assigned. LMs and translation probabilities tables are weighted seeking to optimize their entropy [11] (JURAFSKY; MARTIN, 2008)).

Five types of MT systems were built: *(i)* a baseline, using only in-domain data (*BL*); *(ii)* an unweighted combination of both in and out-of-domain data ($UW$); *(iii)* a combination weighting only LMs (*LMW*); *(iv)* one combination weighting only the phrase translation probabilities table ($PTW$); and *(v)* a system weighting both LMs and phrase translation probabilities table ($FW$). One system of each type was built to translate from German to French and another to translate from German to Italian. The domain of in-

[11] Given a set of in-domain sentences, the more predictable a sentence is, the lower is its entropy according to the LM.

terest was the automobile industry. The evaluation was performed both automatically and manually. The automatic evaluation considered BLEU (PAPINENI et al., 2002), TER (SNOVER et al., 2006) and METEOR (BANERJEE; LAVIE, 2005).

Using the unweighted combination of in and out-of-domain data ($UW$) deteriorated the quality of the baseline (*BL*) translations. The *LMW* strategy yielded improvements only in German to Italian translations. The $PTW$ type of MT system was the one that yielded a more significant quality improvement on the translations. The $FW$ weighting strategy proved not to be worth using, because when comparing to the $PTW$, this strategy resulted in no significant changes in translation quality. The manual evaluation was performed comparing $PTW$ and *BL* systems, and the results indicated that the weighting scheme clearly outperformed the baseline. Although LÄUBLI et al. (2013) makes use of parallel domain-specific corpora, because they are extremely valuable resources for SMT, finding document pairs that are translations of each other, however, is not a simple task.

To overcome this difficulty, USZKOREIT et al. (2010) try to find equivalent or near-equivalent documents in multiple languages. They use a multilingual adaptation of the work of BRODER et al. (1997), whose objective is to identify duplicates and near duplicates in a large amount of documents. To accomplish this task, all documents are represented as a set of randomly selected n-grams (called shingles) and the similarity between any two documents is calculated as the proportion of common shingles.

To take into account the fact that they have to work with many languages, USZKOREIT et al. (2010) propose a strategy that uses an initial, worse, MT system. Initially, the documents are all translated into a common language, using this MT system. Then, two sets of shingles are formed from every document. The first one is called the set of *matching* shingles and is used to build a list of candidate document pairs. A pair of documents is put in this list only if the number of common matching n-grams is equal to or higher than a certain threshold. The second is a set of lower order n-grams and is used to measure the similarity between two documents from a candidate document pair. Unlike BRODER et al. (1997), in this work, the similarity between two documents is measured by using only the IDF feature of the document vectors.

Finally, sentences are aligned from every document-pair and they are used to train an MT system. Interestingly, this algorithm allows for a cyclic training of an SMT system, if the process is repeated. In the experiments, about one billion documents are used as input for their method.

Considering that domain-specific parallel data is a rare resource and that there is a great difficulty for mining it from large corpora, the work of BERTOLDI; FEDERICO (2009) tries to perform SMT domain adaptation using only monolingual data. It uses an SMT system trained on a corpus of another domain to translate the monolingual data, which may be either from the source or from the target language, generating a synthetic bilingual corpus of the domain of interest.

Their experiments were conducted using the United Nations corpus to train the original system and the in-domain monolingual data was extracted from Europarl documents. Although these seem to be quite similar corpora, the differences in the rate of unknown words and in the perplexity of the LMs that may be built from them [12] between both show that this assumption does not hold. The translations were made from Spanish to English. New LMs and translation models were built using the synthetic data and the usage of these were assessed both alone and combined with the ones from the original system. Although

[12]The perplexity of an LM given a document is proportional to its entropy. Therefore, low perplexity scores indicate that the LM describes well the document.

they claim not to use bilingual data, they need a considerable amount of it (around 200 parallel sentences) for tuning the translation models.

The results obtained showed that it is possible to have significant improvements in SMT quality even when only monolingual data is available. Major contributions were made by the domain-specific LMs, while the TMs had minor, but still positive, effects.

## 3.3 Distributional Thesauri Construction

A comparison of several methods for estimating semantically related words is presented in PADRÓ et al. (2014). They build a set of 18 different distributional thesauri, varying the minimum number of times a word has to appear in a context of another one and the similarity measure used for comparing two different context vectors. They are compared against each other and to 2 thesauri built from knowledge bases. Also, they evaluate their effectiveness in performing NLP tasks such as a WordNet Based Synonymy Test (WBST). The WBST is a test for assessing how good a thesaurus is on estimating synonyms extracted from the WordNet. The analyses indicate that the most important parameter is the low frequency threshold, which they varied from 1 to 50, achieving better results with the later.

The efficiency of this kind of process can be enhanced by using a knowledge base. The work of MOHAMMAD et al. (2007), however, observes that for some resource-poor languages this is still particularly hard to do. Their goal is to use one resource-rich language to enhance the estimation of related words in another, resource-poor language. By combining a bilingual dictionary between the resource-poor and the resource-rich languages (respectively, the target and the source languages) and a knowledge base in the rich one, they proved that it is possible to build distributional profiles of concepts from the language with fewer resources.

First, each word in the resource-rich language that is contained in the bilingual dictionary is mapped to its possible senses, according to the knowledge base. Then, by looking up the dictionary, each of these words is also mapped to its translation in the target language. Hence, the words in the resource-rich language work as links between their translations in the other language and their possible senses. A matrix, called the base Word-Category Co-occurrence Matrix (WCCM), is built where any given cell $m_{i,j}$ contains the number of times a word $i$ co-occurs with any word that can be used with the sense $j$. This matrix is used to disambiguate the words that co-occur with the analyzed words and another matrix, the bootstrapped WCCM is built, with $m_{i,j}$ containing the number of times that the word $i$ co-occur with another word that is thought to be used with the sense $j$.

They perform experiments using English as the resource-rich language and German as the resource-poor one. Although the choice of German as the resource-poor language seems contradictory, since it is not really a language with scarce resources, its use was necessary in order to be able to compare the solution with monolingual state-of-the-art methods that use knowledge bases. Tests included ranking word pairs by their similarity and a set of multiple choice questions, where the correct answer is the word that most relates to the question. In both cases, their approach overcame the best monolingual algorithm. In our work, we use a similar approach to seek for equivalent terms in two different languages, building a domain-specific phrase table.

Another approach for improving the construction of thesauri is proposed in CLAVEAU; KIJAK; FERRET (2014). Based on the main idea of exploiting the list of close neighbors from a thesaurus entry, they proposed 3 main contributions. The first contribution resides in examining the reciprocity of the relation (if a word $w_1$ contains a word $w_2$ in its close neighbors list and $w_2$ also has $w_1$ in its list). If the relation is reciprocal, then the words must really be related. The second contribution consists in assigning confidence scores to the thesaurus entries. An entry is probably a good entry if the distance (in number of ranks) with all its close neighbors is similar to the distance between the same words, when observed in all the other entries. The last one is the local reranking algorithm, which swaps the ranks of some of the neighbors, from lists with low confidence scores, based on the probabilities that each neighbor has to be in each rank. Results indicate that the approaches bring significant improvement on the original neighbor lists, when comparing their performances on WBST.

The following Section summarizes the works studied in this Section, along with the main differences and similarities between them and our work.

## 3.4 Comparison

We have presented a number of FC algorithms. The simplest ones, like Fish Search (DE BRA et al., 1994) and Shark Search (HERSOVICI et al., 1998), are based on the similarity of the pages, or some part of it, to a user defined query. In order to take advantage of locality information inside the documents, we use the Shark Search in our experiments, instead of Fish Search.

Also, some more sophisticated algorithms for FC were presented, like the one by CHAKRABARTI; BERG; DOM (1999), based on text classifiers and distillers, and the algorithm based on learning automata, proposed in AKBARI TORKESTANI (2012). Besides, we have discussed works that propose to seek relevant pages by looking at their neighborhood. The one from HSU; WU (2006) does it based on context graphs, built by exploiting the capability of a SE to retrieve backlinks, and the work of LIU; JANSSEN; MILIOS (2006) is based on HMMs, which are constructed by monitoring some browsing sessions of specialist users. We implemented and experimented the one from LIU; JANSSEN; MILIOS (2006), but not the ones from CHAKRABARTI; BERG; DOM (1999) and HSU; WU (2006), because we do not desire to depend on knowledge bases such as the tree of directories from Yahoo! (2012) or on the use of SEs, that can block or change the way they are accessed at any time.

Another approach for constructing comparable corpora that exploits ontologies and SEs is presented in GRANADA et al. (2012). Because of its dependencies on SEs and knowledge bases, we also do not implement this work. The work of TALVENSAARI et al. (2008), in turn, is quite similar to ours, because they apply FC for SMT domain adaptation. In our work, however, we experiment several FC algorithms, build LMs with the collected texts, search for equivalent words (instead of parallel paragraphs), perform the evaluation based on translations of well-structured sentences and investigate whether there is a correlation between the quality of both processes, which they could not assess, since they only used one FC algorithm.

For assessing our SMT systems, we use BLEU (PAPINENI et al., 2002), but a more flexible and complete evaluation, like the one from the work of GIMÉNEZ; AMIGÓ (2006) may be a continuation for the present work. We have presented a methodology for analyzing different error types, in the work of IRVINE et al. (2013), which we could also apply for examining the errors in the SMT process and determining the specific points that need further improvement.

The approach used by USZKOREIT et al. (2010) and BERTOLDI; FEDERICO (2009), that generates synthetic bilingual corpora, could not be applied for generating domain-specific phrase tables, because, in our case, we had to translate sets of very specific sentences and, thus, its performance proved to be unsatisfactory. As observed in the work of LÄUBLI et al. (2013), that compares several strategies for combining in-domain with out-of-domain data, we also will, as a future work, investigate an intelligent way of handling this, by weighting, at least, the phrase table, which showed significant improvements on the other strategies presented.

As explained in Section 4.3.1, we rely on the work of MOHAMMAD et al. (2007) for finding equivalent terms. Unlike them, however, we seek for synonyms that cross the language barrier, which poses a considerable challenge. The observations made in the thorough analysis presented in the work of PADRÓ et al. (2014) may be helpful, especially the variation of the low frequency threshold, that, in their experiments, showed the most significant improvements, when using high values for the threshold.

# 4 CRAWLING FOR SMT DOMAIN ADAPTATION

This Chapter describes the methodology that we used in this work. We divide this description into three parts. In the first part, we give an overview of the whole methodology. The second part explains the crawling stage, describing the focused crawling algorithms applied. Finally, the third part explains the stages that concern domain-specific MT.

## 4.1 Overview

The process starts with the FC stage (Section 4.2). In this stage, we run several FC algorithms, varying their set of seed pages and the languages of the collected documents. For every algorithm, we store all the collected documents, along with their estimated relevance, in order to assess their Average Precision.

The second stage uses the domain-specific crawled corpora for SMT domain adaptation (Section 4.3). We begin by training an SMT system with generic training data. Then, we preprocess all the collected pages to allow them to be used as additional training data, for performing domain adaptation. The adapted system is used to translate a set of domain-specific sentences. Finally, we perform MT evaluation, in order to assess the quality of the SMT systems we have trained.

Figure 4.1 illustrates the whole methodology, in a high-level view. Recall from Section 1 that the main goal of this work is to assess whether FC is a good alternative for performing SMT domain adaptation and if there is a correlation between the quality of FC algorithms and of the corresponding domain-specific SMT systems. Thus, our methodology consists in experimenting with both processes and comparing their quality.

## 4.2 Crawling for Comparable Corpora

The crawling stage is responsible for acquiring comparable corpora. The subject of the collected texts is supposed to be related to the domain of interest. Also, their language should be either the target or the source language that will be used in the SMT experiments.

Since we needed to experiment with several types of FC, we have built a generic crawler that can be easily extended, according to the necessity, which is one contribution of this work. The only thing that one needs to do in order to extend it is to write a class that overrides a method for assigning a priority score for any given document pointed by a link[13].

[13]The crawler's source code and the necessary resources can be found in `http://inf.ufrgs.br/~brlaranjeira/crawler/getcrawler.html`

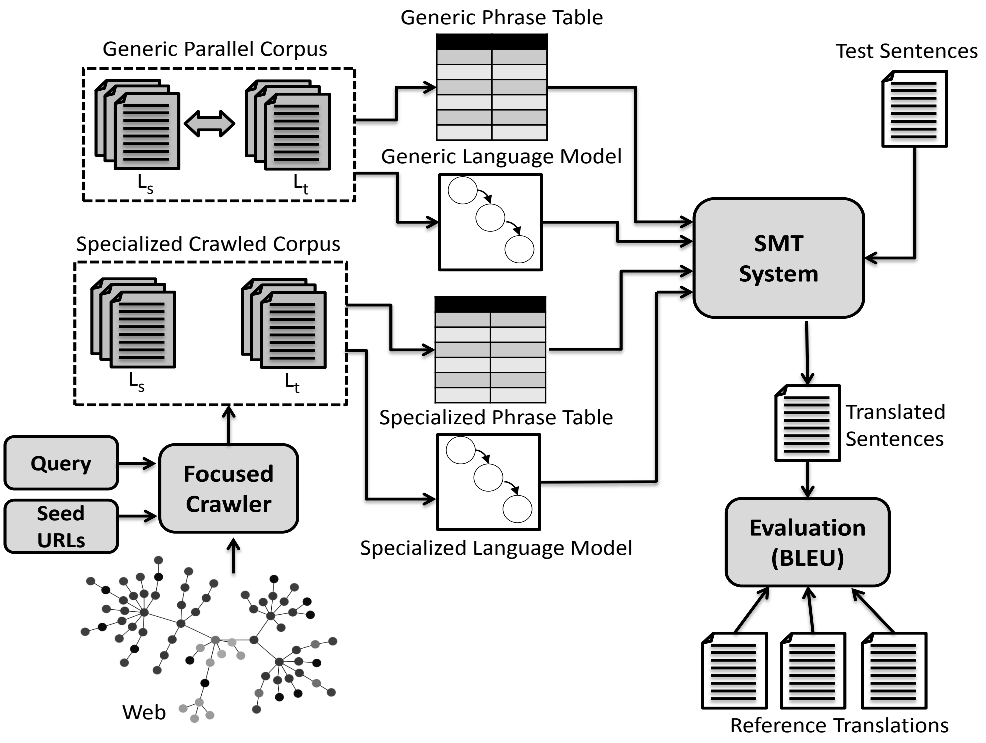


Figure 4.1: Overview of the whole process.

Before actually initiating the crawl, it is necessary to have some resources that the crawler uses for focusing the crawling process and evaluating its quality. One of these resources is a file, or a set of files, containing IDF values. The IDF of a word or n-gram expresses how rare it is in a given collection and its value the log of the ratio between the number of documents in the collection and the number of documents in which the word or n-gram occurs, as formalized in Equation 4.1, where *D* is the collection and *w* is the word or n-gram analyzed.

$$idf(w,C) = \log \frac{||C||}{||\{c \in C, w \in c\}||} \tag{4.1}$$

These values should be calculated from a generic collection of documents (*e.g.*: newspaper texts), because they should describe how frequent the words are usually employed in a language and, thus, also assigning higher importance to domain-specific terms and phrases than to other ones. Some algorithms, like the MWE FC, introduced in Section 4.2.2, may also need values of some n-grams PMI or any other useful frequency information available.

A set of seed URLs must be specified so that the crawler knows where to start traversing the web graph. As seen in Section 2.1.1, the Radius-One Hypothesis states that the more related a topic is to a page, the higher is its probability to point to other documents related to the same topic. Hence, we may say that the more the seed pages are related to the domain of interest, the higher is the probability of the FC process to have a good quality, measured either by its Harvest Rate or Average Precision.

Another input that should be provided to the crawler is a query document. The query will be used for two distinct purposes. It will guide, or focus, the crawl. Based on the

Radius-One Hypothesis, most algorithms use the query to estimate how relevant a given page (or a fragment) is and use this value as the probability that it points to relevant documents. The other purpose is to assess the quality of the process. The evaluation is usually done based on the cosine similarity between the query and every collected document, using metrics like the Harvest Rate or the Average Precision. Moreover, depending on the focusing strategy used, it is possible to assign two different queries, one for each purpose. It is essential, for a fair comparison, that the query used for the evaluation is the same across all the compared FC algorithms. Both queries may be represented by a set of words or URLs. In the later case, the words in the documents pointed by them are used to compose the query.

To compute the similarity between the query and a page, we first convert the query into a bag of words. Instead of telling how many times each word appears in the query, it contains their respective TF-IDF values. When a page is collected, the same process is applied to it. We also try to extract information of images in the HTML page, by getting the *alt* attribute, when they are present in the *img* tags. This attribute has the function of replacing the image with a text, when there is an issue that does not allow the browser to load it. Hence, there is a chance of this text to properly describe the content of the image. Having both query and page represented in this model of document, it becomes quite trivial to use TF-IDF values to compute their cosine similarity.

We need the user to tell us the most probable encoding in which the crawler will find the pages to be collected, because finding the right encoding was a problematic issue. Many HTML pages omit the *charset* attribute in the *meta* tags and algorithms we have experimented for its automatic detection also did not work well [14]. All these specifications must be included in a configuration file that is given as input for the crawler. This file also ought to inform parameters like the output directory, the number of concurrent threads, the number of pages to be collected, and the FC algorithm to be used.

Figure 4.2 summarizes the logical structure of how our FC works. After the *Config* module has loaded all the user defined options, the *Main* method starts processing the URLs received from the *URL Frontier Handler*, that manages an URL frontier by sorting it in a way that its component URLs follow the focusing algorithm specified by the user. The scores assigned to every page are normally assigned by the *Document Similarity* module, which employs the cosine similarity metric. Different FC strategies, however, may easily use any other method, but, as we have explained, the standard calculation is used for evaluating all the algorithms. When a page is collected, the estimated standard relevance of the document and the partial average relevance of all collected documents are both saved in a log file for further analyses.

Before adding any URL to the frontier, the crawler needs to be aware of its permission to do so, according to the Robots Exclusion Protocol (JHA et al., 2014). To ensure that this permission is respected, the *Robots Analyzer* reads the *robots.txt* file, present in the root of the host server. If the file tells the crawler not to follow a path that contains the URL, then it is not added to the frontier and, therefore, will not be accessed. Another criterion that a URL has to satisfy to be added to the frontier is that its domain [15] needs to be contained in a set of domains specified by the user. These domains are used to keep the crawler collecting pages from the intended language. This approach may cause some false negatives, excluding pages that are from the language but with the URL in another

[14]Most of times it detected UTF-8 files, even when they had another encoding, like ISO-8859-1.

[15]Here, we do not use the term domain as a synonym of topic, but as a component of the host of the URL, like, *e.g.*, ".com", ".br" or ".pt".

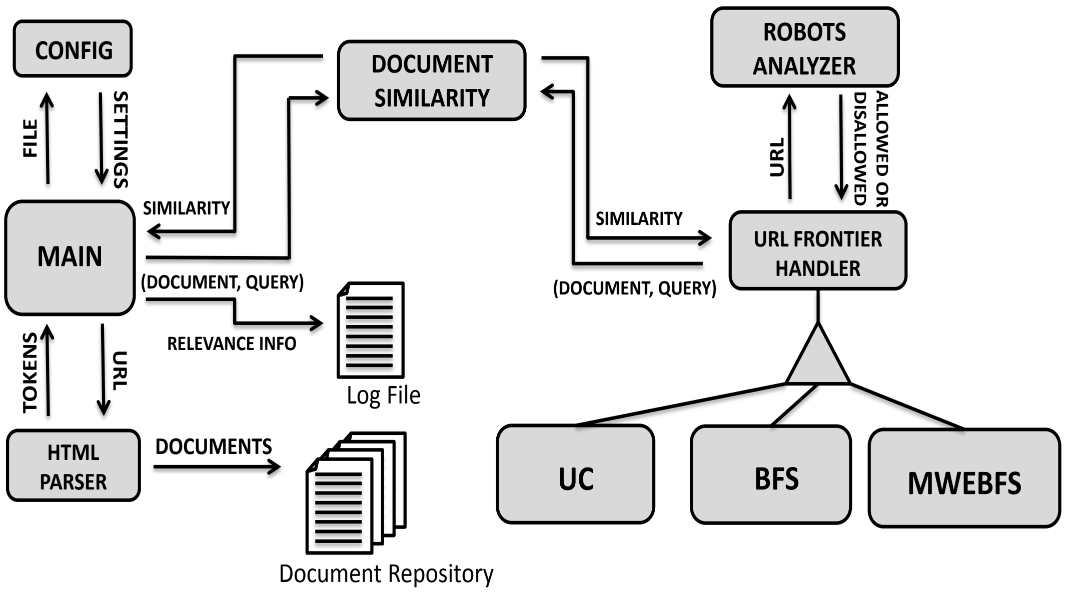


Figure 4.2: Logical structure of the focused crawler.

domain, and also a small number of false positives, by allowing pages whose URL is within the specified domain but written in another language. Nonetheless, this domain filter was a relatively successful strategy for most pages.

The *Html Parser* receives an URL as input, establishes a connection to the server, resolves its redirecting issues and parses the corresponding HTML file into plain text that is stored in a document repository. This text is split in tokens which, depending on the user configuration, may be stemmed, using the appropriate stemmer for the current language, or simply kept in its raw form.

Algorithm 1 details the sequential logic used in our FC. An interesting detail can be observed in line 14. When the crawler finds a URL that has not been visited, but it is already in the frontier, it is necessary to decide what score the URL is going to have. This policy is also customizable, and it is desirable that the implementer of any FC algorithm considers this to be an essential question, because in some cases it changes the crawler behavior.

We have written a set of FC algorithms, with some represented in Figure 4.2. Some of them are strategies that can be found in the literature and are described in Sections 2 and 3. Those are *Best-N-First*, with varying sizes of *N*, *Shark-Search* (HERSOVICI et al., 1998) and the *HMM Based Focused Crawler* (LIU; JANSSEN; MILIOS, 2006), in addition to a universal unfocused crawler, which follows a breadth-first search. Others, in turn, are novel approaches, based on some standard FC algorithms. We describe the crawling strategies proposed in this work in the following Sections.

To allow an easy configuration of the crawler, we have developed a simple web-based application for generating the config files with a graphical interface. In Figure 4.3, we show a screen capture of the system[16].

[16]The generator system is available at `http://inf.ufrgs.br/~brlaranjeira/crawler/`.

**Algorithm 1** Focused Crawling Algorithm

```
1:  procedure FOCUSED CRAWLING(seeds,limit,domain_list)
2:      $frontier \leftarrow seeds$
3:      while SIZE($frontier$) $> 0$ & SIZE($collected$) $< limit$ do
4:          $URL \leftarrow$ HEAD($frontier$)
5:          ADD($visited, URL$)
6:          $text \leftarrow$ EXTRACT_DOCUMENT($URL$)
7:          $links \leftarrow$ EXTRACT_LINKS($URL$)
8:          for all $tmp\_link \leftarrow links$ do
9:              if ROBOTS($tmp\_link$) & GET_DOMAIN($tmp\_link$) $\in domain\_list$ then
10:                 if $tmp\_link \notin visited$ then
11:                     $score \leftarrow$ GET_SCORE($text, tmp\_link$)
12:                     if $tmp\_link \in frontier$ then
13:                         $old\_score \leftarrow$ GET($frontier, tmp\_link$)
14:                         $new\_score \leftarrow$ COMBINE($old\_score, score$)
15:                         ADD($frontier, tmp\_link, new\_score$)
16:                     else
17:                         ADD($frontier, tmp\_link, score$)
```

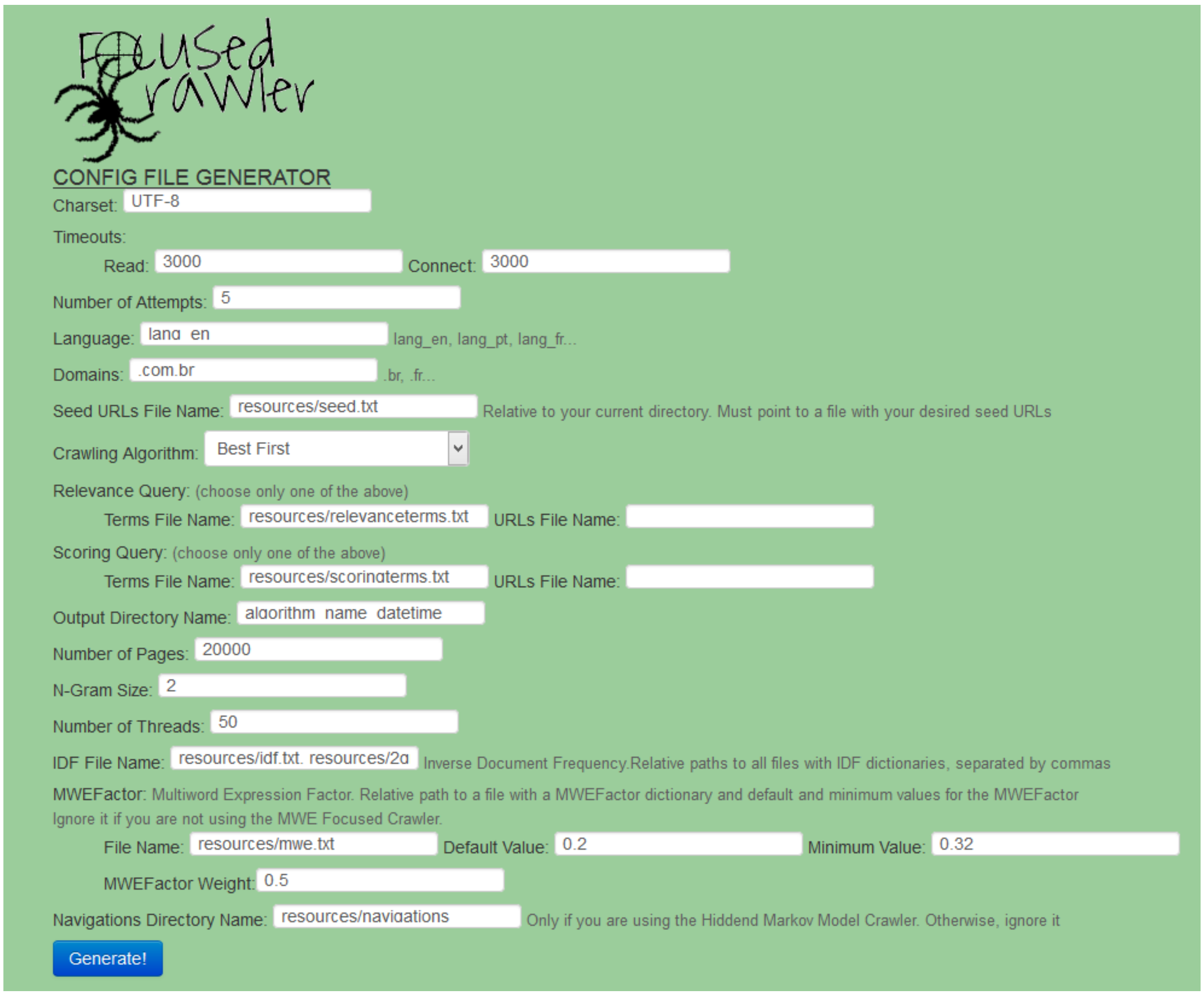
### 4.2.1 N-Gram Based Focused Crawlers

One of the variations we propose concerns the way that texts from pages and queries are interpreted when they are parsed to build the bag of words representation. Instead of using a single word as the smallest textual unit, we use n-grams. These n-grams must have a fixed size (measured in number of words). The idea for using them is to try to give more importance to phrases and compound words, *e.g.*, "phrasal verbs", when the n-gram size is 2, or "statistical machine translation", when using size 3. This is specially interesting in domains where a large proportion of terminology is composed by more than one word.

We can apply this idea to almost every FC algorithm. The main difference is that documents and queries used by the method that guides the crawler must be built by observing word sequences instead of single words. It is possible to use both approaches, interpolating the single words model and the one we proposed, with n-grams. For this, it is only necessary to specify the weight for the n-gram approach in a linear combination. This weight is used to define the importance of both approaches in the score assigned to an URL and must be a real number between 0 and 1. Equation 4.2 formalizes how this weighting approach works, where $u$ represents a URL, $q$ a query and $\alpha$ the weight. The *score* function represents the final score assigned to $u$ in the URLs frontier and *ngram_score* and *word_score* are the functions that measure the importance of $u$, according to $q$, considering, respectively, n-grams and words as the smallest textual units.

$$score(u,q,\alpha) = \alpha \times ngram\_score(u,q) + (1-\alpha) \times word\_score(u,q) \tag{4.2}$$

We have adapted the *Best-N-First* and the *Shark-Search* algorithms to work with n-grams. When computing the *ngram_score*, the algorithm needs to be aware of the IDF values of every observed n-gram. This requires parsing the whole generic collection used for building the IDF vector again, in order to analyze frequencies of n-grams. However, the evaluation of the quality of the algorithms still needs to be done using documents



CONFIG FILE GENERATOR

Charset: UTF-8

Timeouts:

Read: 3000 Connect: 3000

Number of Attempts: 5

Language: lang_en lang_en, lang_pt, lang_fr...

Domains: .com.br .br, .fr...

Seed URLs File Name: resources/seed.txt Relative to your current directory. Must point to a file with your desired seed URLs

Crawling Algorithm: Best First

Relevance Query: (choose only one of the above)

Terms File Name: resources/relevanceterms.txt URLs File Name:

Scoring Query: (choose only one of the above)

Terms File Name: resources/scoringterms.txt URLs File Name:

Output Directory Name: algorithm_name_datetime

Number of Pages: 20000

N-Gram Size: 2

Number of Threads: 50

IDF File Name: resources/idf.txt, resources/2g Inverse Document Frequency.Relative paths to all files with IDF dictionaries, separated by commas

MWEFactor: Multiword Expression Factor. Relative path to a file with a MWEFactor dictionary and default and minimum values for the MWEFactor Ignore it if you are not using the MWE Focused Crawler.

File Name: resources/mwe.txt Default Value: 0.2 Minimum Value: 0.32

MWEFactor Weight: 0.5

Navigations Directory Name: resources/navigations Only if you are using the Hiddend Markov Model Crawler. Otherwise, ignore it

Figure 4.3: Screen capture of the application for generating the config files.

represented as single words, for a fair comparison.

As the size of the vocabulary or of the n-grams increases, the number of distinct n-grams found in the collection results in a combinatory explosion. Thus, keeping all these values in memory would certainly cause a stack overflow and seeking them in disk every time that they are needed would be too time consuming. To overcome this problem, we used a simple pruning strategy. When building the file containing the IDF values for the crawler, we treated every n-gram that occurred less than a user-defined threshold as if it did not exist in the collection. Hence, in the crawling process, they were treated just like hapaxes, (*i.e.*, words that are seen only once). The IDF value of hapax n-grams and words can be easily computed if the number of documents of the collection is known, just by calculating the IDF of an arbitrary word that happens to occur only once in it.

### 4.2.2 MWE Focused Crawler

In order to further improve our n-gram based approach, we sought for frequency information that could be used to measure how much information a given n-gram carries. Multiword expressions are, by definition, n-grams whose meaning must be interpreted as a unit, usually transcending the one that could be deduced by combining the sense of their component words. Hence, we found that an appropriate way to say whether [17] an n-gram is a MWE would also be an indicative of the n-gram's usefulness in the document representation.

A simple way to estimate if an n-gram corresponds to a unit of meaning is to compare the frequency of the whole n-gram with that of its component words. We chose to use the normalized PMI of an n-gram as the association metric to perform this analysis.

We say, then, that the PMI value of an n-gram is called its *Multiword Expression Factor*, or simply *MWE Factor*. The type of FC that uses this factor is called a *MWE Focused Crawler*, or *MWE FC*. Just like in the n-gram approach, we also needed the IDF values for single words and n-grams and a user-defined weighting value. Besides that, it was also needed to consider the MWE Factor when estimating the relevance score of a page. Hence, we have adapted the classical TF-IDF model.

The model used in a MWE FC is the TF-IDF-MWE, with the third factor corresponding to the n-gram MWE Factor. Instead of simply multiplying the TF factor by the frequency of the n-gram, it is multiplied by a more complex factor, the IDF_MWE factor, as shown in Equation 4.3, where $n$ and $D$ are respectively the n-gram and the document being analyzed, $C$ is the generic collection analyzed and $TF$ is a function that returns the observed frequency of $n$ in $D$. This factor is composed by the IDF and the MWE Factor of the n-gram. The value of IDF ranges from 0 to $\log|C|$ and the MWE Factor, from 0 to 1. To smooth this difference, the MWE Factor is multiplied by $\log|C|$, to make both factors to range between the same values. Still, a constant must be defined by the user, to balance the importance of each factor. Equation 4.4 formalizes the idea of the IDF_MWE factor, where $\beta$ is the weight we have just explained and *IDF* and *MWE* functions calculate the IDF and the MWE Factor of $n$ in $C$, respectively.

$$TF - IDF - MWE(n,D,C,\beta) = TF(n,D) \times IDF\_MWE(n,C,\beta) \tag{4.3}$$

$$IDF\_MWE(n,C,\beta) = (1-\beta) \times IDF(n,C) + \beta \times \log|C| \times MWE(n,C) \tag{4.4}$$

Since we had to apply a pruning strategy to the n-grams IDF value, due to their exponential growth, it seemed quite logical that the same problem would arise with the MWE Factor. But unlike the former case, where the pruning could be done just based on the frequency, because the IDF value can be directly inferred from it, it was necessary to properly calculate the PMI value (as described in Section 2.2.4) and perform the pruning based on the result. The n-grams with the PMI value lower than a threshold, defined by the user in the config file, are discarded by the crawler when it is parsing the file containing all these values. Again, similar to the treatment used for finding the IDF of pruned n-grams, we may consider the n-grams whose MWE Factors were pruned equals to the ones that did not occur even once in the corpus, because neither of them tend to be a MWE. In these cases, the MWE Factor is defined by the user as an arbitrary low value close to zero.

## 4.3 Domain-Specific Statistical Machine Translation

After the comparable corpora with domain-specific texts in the source and the target languages are collected the SMT stage can take place to translate the text documents. This

[17] Or how much, since we do not consider this to be a binary concept.

stage is composed of three main steps: (*i*) preprocessing, (*ii*) training the SMT system and (*iii*) applying the SMT system.

The preprocessing started by concatenating all files into one and converting it to a common encoding. The resulting file was used as input in a Python script, to split the input file into sentences, following a set of rules from the text language. The duplicated sentences and the ones that were not considered from the intended language were removed. Finally, the sentences were tokenized using the appropriate rules for each language.

Then, every sentence, from source and target languages, is treated as an individual document and is indexed by a SE. The reason for indexing the sentences is to reduce the time in the part of the process of training the SMT system that builds a bilingual domain-specific lexicon, avoiding the necessity of reading the whole set of sentences numerous times to seek for sentences containing the sought term.

The training and the application of the SMT systems are described in details in the Sections 4.3.1 and 4.3.2.

### 4.3.1 Training the SMT System

For building the SMT systems, we first constructed training data from generic parallel corpora. Then, for adapting it to a specific domain we added the in-domain training data collected with the FC algorithms, in both source and target languages. Building LMs, which are only necessary in the target language, because its purpose is to make the output text more fluent, involves calculating the sequences of words that are more likely to happen. The other part, which is more complex, is when the parallel data is needed, in order to build the table of translation probabilities.

Generating an LM is quite simple, if there is access to an appropriate tool. After the corpora have been preprocessed, the only thing needed is to provide natural language texts as input to the system and to specify the order of the n-grams that will compose the LM. The order of an LM is the maximum size of the n-grams computed while building the table of probabilities of n-gram occurrences. If the order of the LM is four, for example, the three last words are observed, in order to estimate the probabilities of the fourth word, given the previous three words.

Besides building an LM model with the texts collected by our FC, we need to combine it with another LM. Although this other LM is built with generic texts, its usage is important due to its larger size. Our LM, in turn, is important so that the combined model is adapted to the domain-specific vocabulary.

In addition to using these LMs, we also tried to find cross-lingual, domain-specific, equivalent terms, by examining the corpora collected in both languages. We did this seeking to improve our SMT's phrase tables, for them to be more capable of properly translating domain-specific terms and phrases.

To construct this bilingual lexicon, we have used an approach based on the work of MOHAMMAD et al. (2007). Having parallel corpora, aligning words is a relatively simple task. But if the corpora are comparable, this becomes substantially harder, because there is no information that says which sentences are translations of each other. For both collected corpora (in source and target languages) we extract a set of important terms. They were ranked by multiplying their frequencies (term-frequency) in the collected corpus by their IDF in the generic ones analyzed previously. Then, we tried to align these terms with their respective translation in the other corpus. For this, we built distributional profiles for each of the extracted terms, analyzing the sentences retrieved by the SE when submitting queries containing them. The vector features are represented by the same set of words in both source and target languages. Each feature is composed by a set of cross lingual synonyms, *e.g.*, “trabalho”, in Portuguese, and “work”, in English. These synonyms are found using a bilingual generic lexicon. Hence, some words may be found in more than one feature, because they may refer to different senses. The Portuguese word “banco”, for example, can be found along with the English word “bank” in one feature and with “bench” in another one, since both are possible English translations of the Portuguese term.

The value of each of these features of the distributional profile of the terms is calculated by the number of times that the word co-occurs with any of the synonym words that compose the feature times its IDF. After these profiles are constructed, it is possible to find cross-lingual equivalent terms, because their profiles in both source and target languages are built considering the same features.

This is calculated between every pair of source-target terms. Thus, we have a total of $N^2$ comparisons, where $N$ is the number of important terms we have extracted from each collected corpus. Then, every pair of words were filtered according to their computed similarity value. This similarity was, then, used as the probability of the word pair entry in the phrase table.

In order to properly combine both generic and domain-specific LMs and phrase tables, we must be aware of what are the optimal weights for the features they represent to the SMT algorithm. We have tested all SMT algorithms without weighting the data and also weighting the features with the Minimum Error Rate Training (MERT) algorithm.

### 4.3.2 Using the SMT system

In this step, we perform automatic translation of the input sentences from the source to the target language and their evaluation. The first is almost completely automatic, while the second, in turn, is highly dependent on the intervention of humans that are linguists or experts in the domain of interest.

Before translating, we must select some domain-specific sentences to be used as input in the trained SMT system. For this, we use the index previously constructed. We extract some important terms, manually filter them whether they are domain-specific or not, combine them in pairs and triples and submit these combinations to the SE. Some of the retrieved sentences (starting from the first returned results) from each query are manually chosen to be the sentences that are decoded by the SMT system.

Then, all the selected sentences are translated by all the SMT systems.The output sentences are stored in log files that are further used in the evaluation of the corresponding SMT systems.

The evaluation is a process that depends heavily on human intervention. Domain specialists or linguists are needed to manually translate the same sentences that the SMT systems translate. These translations are considered as gold-standard, or reference translations. In the evaluation of an SMT system, we apply the BLEU metric, which compares an automatically translated sentence to a set of manually built reference translations (PAPINENI et al., 2002). The metric outputs a series of results, each of them considering the precision of the system in a different n-gram granularity. The results of order 1, for example, only consider whether the words of the automatic translation are the same as those in the gold-standard. In the results of order 2, the quality is measured by the proportion of common 2-grams. Also, there is a final result that combines them all in a logarithmic scale. This scale is used because these scores tend to greatly decrease as the order of the n-gram grows.

The evaluation of an SMT system using BLEU, however, also has some flaws. If the system generates, for example, the sentence "*John is faster than Michael.*", while the correct translation should be "*Michael is faster than John.*" , the BLEU score of order 1 of this system is perfect, even knowing that the sense of the sentence has changed. Analogously, the same thing can happen when using BLEU for analyzing more coarse-grained n-grams. We attenuate this problem by considering multiple reference translations. A fully manual evaluation of SMT systems' output, however, would be too time-consuming and is out of the scope of this work.

After both FC and SMT processes are complete, we analyze and compare them, in order to find the correlation between the obtained values. This analysis is described in detail in the Chapter 5.

# 5 EXPERIMENTS AND RESULTS

In this Chapter, we describe all our experiments in detail. The first two sections detail our experiments on FC and domain adaptation for SMT, respectively. Both start by specifying the experimental setup followed by the results. Later, in Section 5.3, we discuss the obtained results.

## 5.1 Focused Crawling

In this Section we describe in details the parameters used to run our Focused Crawling experiments and the obtained results.

### 5.1.1 Experimental Setup

For the FC experiments, we have considered a universe of six distinct base algorithms and the topics we used were *dermatology* and *software engineering*. These domains were chosen by the convenience of finding gold-standard data for the SMT stage. For the dermatology domain, we were assisted by Linguistics students and professional translators. The software engineering domain was chosen because the authors could manually translate the set of sentences from the source language to the target language. The languages adopted were English and Brazilian Portuguese.

The variation of parameters for the FC base strategies resulted in a set with 12 different algorithm variations. This parameter variation was made for the same base algorithms to be experimented considering different granularity sizes for the document components. When the chosen granularity is an n-gram, the crawler needed to have some extra resources, such as the IDF values, which in turn required information on their frequency of occurrence in the corpus, or their MWE Factor, which depends on the calculation of the PMI values of the respective n-grams.

In Table 5.1, we can see all the base algorithms and their variations. The first column identifies the algorithm, the second column lists the variations applied, and the third column shows a mnemonical representation, which will be used as a reference for them in the remainder of this thesis.

Another configuration that the FC requires is the set of seed URLs. Although we experimented them with two distinct sets of URLs, we did not consider it as a variation of the algorithm, since it only consisted in a difference in its initial conditions and not in the crawler strategy for organizing the frontiers queue.

One of the two sets we have used was considered the set of *good seeds* and the other one was composed by the *bad seeds*. Both sets of seed URLs were manually chosen with queries submitted to an SE. Each domain of interest had a set of good seeds, that

Table 5.1: List of the FC base algorithms, their variations and the respective mnemonics.

| Base Algorithm | Variation | Mnemonic |
|---|---|---|
| Universal | Universal | UC |
| Best-First Search | Regular Algorithm | BFS-1G |
| | Variation with 2-Grams | BFS-2G |
| | Variation with 3-Grams | BFS-3G |
| Shark Search | Regular Algorithm | SS-1G |
| | Variation with 2-Grams | SS-2G |
| | Variation with 3-Grams | SS-3G |
| MWE Best-First Search | With 2-Grams | MWEBFS-2G |
| | With 3-Grams | MWEBFS-3G |
| MWE Shark Search | With 2-Grams | MWESS-2G |
| | With 3-Grams | MWESS-3G |
| HMM Based | HMM Based | HMM |

contained URLs which were related to it. Unlike the set of good seeds, the set of bad seeds was the same across the topics and the component URLs were related to sports and education. The set of bad seeds is used to check whether the algorithms are robust enough to move away from the seed pages and towards in-domain pages in the web graph.

Tables 5.2 and 5.3 show all URL sets for, respectively, English and Portuguese. Both sets of good seeds, one for every domain, are represented in each Table. These sets were manually selected using standard web SE. We tried to maximize the relevance to the target domains (good seeds) and to select websites unrelated to the target domain (bad seeds).

The FC algorithms we tested are:

- ***UC:*** representing the absence of a strategy for focusing the crawling procedure. It consists in following the links in the same order that they are found, in a *breadth first search*.

- ***BFS-1G, BFS-2G, BFS-3G:*** variations of the *Best-First Crawler* base algorithm, described in Table 5.1. $BFS-1G$ is the regular Best-First algorithm, as described in LIU (2011). $BFS-2G$ and $BFS-3G$ are its variations that consider n-gram of size 2 and 3, respectively, as the textual unit (as described in Section 4.2.1).

- ***SS-1G, SS-2G, SS-3G:*** based on the *Shark Search* algorithm (HERSOVICI et al., 1998). The regular algorithm is the $SS-1G$, and its variations using n-grams, such as the *Breadth First Search* based algorithms, are the $SS-2G$ and the $SS-3G$.

- ***MWEBFS-2G, MWEBFS-3G:*** variations of the *MWE Focused Crawler*. Like explained in Section 4.2.2, they add the *MWE Factor* to the $BFS-2G$ and the $BFS-3G$ algorithms, allowing the features of vectors that represent the documents to try to consider how meaningful the described n-grams are.

- ***MWESS-2G, MWESS-3G:*** the *MWE Factor* is also added to modify the $SS-2G$ and the $SS-3G$ algorithms. Just like the $MWEBFS-2G$ and the $MWEBFS-3G$ do to the $BFS-2G$ and to the $BFS-3G$, respectively, the $MWESS-2G$ and the $MWESS-3G$ change the $SS-2G$ and the $SS-3G$ to consider this factor.

- ***HMM:*** the HMM based focused crawler, described in the work of LIU; JANSSEN; MILIOS (2006). For monitoring the user sessions, we have developed an extension

Table 5.2: List of all the URLs that compose all three seed sets for the English language.

| Bad Seeds |
|---|
| `http://msn.foxsports.com/` |
| `http://www.bbc.co.uk/sport/0/` |
| `http://sports.yahoo.com/` |
| `http://www.education.com/` |
| `http://efareport.wordpress.com/` |
| `http://educationusa.org.br/v2/` |
| **Good Seeds (Dermatology)** |
| `http://www.eadv.org/` |
| `http://www.iacdworld.org/home-in.htm` |
| `http://www.sidnet.org/` |
| `http://www.aad.org/` |
| `http://cshm-schm.ca/` |
| `http://emedicine.medscape.com/dermatology` |
| `http://telemedicine.org/stamford.htm` |
| `http://www.dermnetnz.org/` |
| `http://www.medicalnewstoday.com/sections/dermatology/` |
| **Good Seeds (Software Engineering)** |
| `http://en.wikipedia.org/wiki/Software_engineering` |
| `http://en.wikipedia.org/wiki/Software_engineer` |
| `http://www.sei.cmu.edu/` |
| `http://www.computer.org/portal/web/swebok` |
| `http://www.tutorialspoint.com/listtutorials/ software-engineering/1` |
| `http://computingcareers.acm.org/?page_id=12` |
| `https://www.udacity.com/courses#!/software-engineering` |

for the *Google Chrome* browser, and the user only had to mark the relevant pages as favorites and use its output text as input for the crawler training. Figure 5.1 shows a screenshot of the extension. The text contained in the dialog is what must be in the file used to train the crawler. Each line of the text file contains three values, separated by commas. The first denotes whether the line refers to a relevant URL or not, the second represents the id of the browser tab that the user was using, in order to build the navigation graph and the third value is the visited URL.

Except for the *UC*, all these strategies require a set of query documents that characterize the domain. The query is used to score URLs in the algorithms. The queries were constructed with the content present in the Wikipedia article describing the domain of interest.

The IDF and the PMI values were extracted from generic collections from each language experimented. For English we have used a corpus containing the texts from the Los Angeles Times[18] of the year 1994 and from the Glasgow Herald[19] of 1995. For Portuguese, we analyzed the texts from Folha de São Paulo[20], of the years 1994 and 1995.

[18]`http://www.latimes.com`
[19]http://www.heraldscotland.com
[20]http://www.folha.uol.com.br

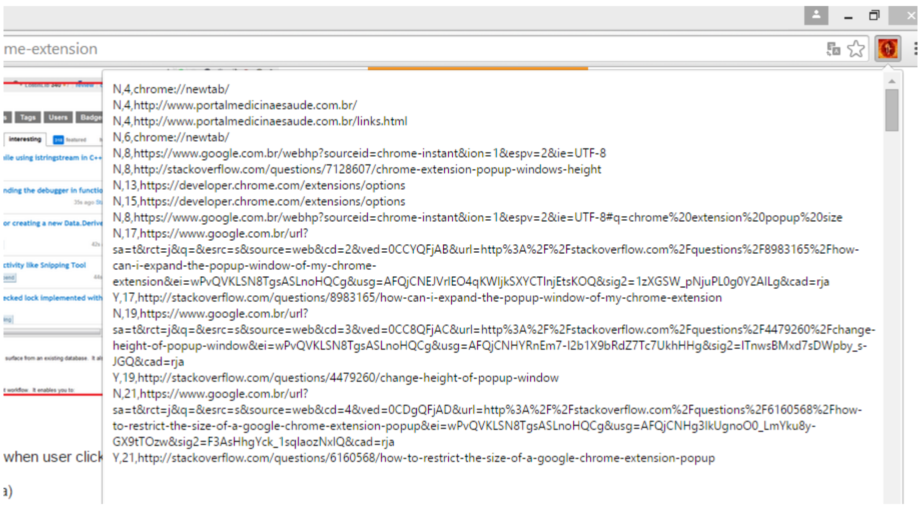
Table 5.3: List of all the URLs that compose all three seed sets for the Portuguese language.

| **Bad Seeds** |
|---|
| `http://www.gazetaesportiva.net/` |
| `http://globoesporte.globo.com/` |
| `http://esporte.uol.com.br/` |
| `http://www.educacao.sp.gov.br/` |
| `http://www.mec.gov.br/` |
| `http://www.estadao.com.br/educacao/` |
| **Good Seeds (Dermatology)** |
| `http://www.virtual.epm.br/cursos/dermabas/frame.htm` |
| `http://www.dermatologia.net/novo/base/index.shtml` |
| `http://pt.wikipedia.org/wiki/Dermatologia` |
| `http://www.sbd.org.br/` |
| `http://www.anaisdedermatologia.org.br/public/default.aspx` |
| `http://protetoresdapele.org.br/` |
| `http://www.enago.com.br/journal/Anais-Brasileiros-de-Dermatologia-402/` |
| `http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0365-05962010000100008` |
| **Good Seeds (Software Engineering)** |
| `http://pt.wikipedia.org/wiki/Engenharia_de_software` |
| `http://engenhariadesoftware.blogspot.com.br/2007/02/o-que-engenharia-de-software.html` |
| `http://www.devmedia.com.br/revista-engenharia-de-software-magazine` |
| `http://www.inf.ufg.br/engenharia-de-software` |
| `http://www.pronus.eng.br/` |
| `http://www.engenhariae.com.br/guia-de-engenharia/o-que-e-a-engenharia-de-software/` |
| `http://lens-ese.cos.ufrj.br/ese/index.php?lang=pt_br` |
| `http://engenhariasoftware.wordpress.com/` |

We used the TextNSP tool (BANERJEE; PEDERSEN, 2003), written in Perl, to perform these analyses.

The FC code was fully written in Java. For parsing the HTML files found during the crawl, we have used a modified version of the HTML Parser library (Oswald, Derrick, 2006). The modifications we have made proved to be necessary, because the original library faced problems while parsing comments in the HTML code. Although their code is successful to parse documents that follow what is defined in the W3C specifications for the HTML, many web pages in the web are not according to the standards, mainly when it does not imply any kind of error message on web browsers. Hence, we have modified this code to be more adapted to the realistic HTML syntax used, allowing for a more flexible parsing of comments.

The library for stemming the words found in the documents was the Snowball stemmer (PORTER, 2009). Stemming is the process of normalizing the variations of a given word, by reducing them to their radical. The words *"calculating"* and *"calculator"* can be



```
N,4,chrome://newtab/
N,4,http://www.portalmedicinaesaude.com.br/
N,4,http://www.portalmedicinaesaude.com.br/links.html
N,6,chrome://newtab/
N,8,https://www.google.com.br/webhp?sourceid=chrome-instant&ion=1&espv=2&ie=UTF-8
N,8,http://stackoverflow.com/questions/7128607/chrome-extension-popup-windows-height
N,13,https://developer.chrome.com/extensions/options
N,15,https://developer.chrome.com/extensions/options
N,8,https://www.google.com.br/webhp?sourceid=chrome-instant&ion=1&espv=2&ie=UTF-8#q=chrome%20extension%20popup%20size
N,17,https://www.google.com.br/url?
sa=t&rct=j&q=&esrc=s&source=web&cd=2&ved=0CCYQFjAB&url=http%3A%2F%2Fstackoverflow.com%2Fquestions%2F8983165%2Fhow-
can-i-expand-the-popup-window-of-my-chrome-
extension&ei=wPvQVKLSN8TgsASLnoHQCg&usg=AFQjCNEJVrlEO4qKWljkSXYCTInjEtsKOQ&sig2=1zXGSW_pNjuPL0g0Y2AILg&cad=rja
Y,17,http://stackoverflow.com/questions/8983165/how-can-i-expand-the-popup-window-of-my-chrome-extension
N,19,https://www.google.com.br/url?
sa=t&rct=j&q=&esrc=s&source=web&cd=3&ved=0CC8QFjAC&url=http%3A%2F%2Fstackoverflow.com%2Fquestions%2F4479260%2Fchange-
height-of-popup-window&ei=wPvQVKLSN8TgsASLnoHQCg&usg=AFQjCNHYRnEm7-I2b1X9bRdZ7Tc7UkhHHg&sig2=ITnwsBMxd7sDWpby_s-
JGQ&cad=rja
Y,19,http://stackoverflow.com/questions/4479260/change-height-of-popup-window
N,21,https://www.google.com.br/url?
sa=t&rct=j&q=&esrc=s&source=web&cd=4&ved=0CDgQFjAD&url=http%3A%2F%2Fstackoverflow.com%2Fquestions%2F6160568%2Fhow-
to-restrict-the-size-of-a-google-chrome-extension-popup&ei=wPvQVKLSN8TgsASLnoHQCg&usg=AFQjCNHg3lkUgnoO0_LmYku8y-
GX9tTOzw&sig2=F3AsHhgYck_1sqlaozNxlQ&cad=rja
Y,21,http://stackoverflow.com/questions/6160568/how-to-restrict-the-size-of-a-google-chrome-extension-popup
```

Figure 5.1: Screenshot of the extension developed for monitoring the user sessions.

reduced to the same radical *"calculat"*, for example. The stemmed words were used only to build the vectors representing the documents. The files saved in the document repository, for the SMT procedures, contained the words in their raw forms.

For the *HMM* algorithm, we have used the Java Application Programming Interface (API) of the software *Waikato Environment for Knowledge Analysis* (WEKA) (HALL et al., 2009) for applying the XMEANS algorithm (PELLEG; MOORE et al., 2000). The *Jahmm* implementation of HMM (FRANCOIS, 2010) was used for building the HMM and also for estimating the hidden state, given the cluster of the observed page, determined by the WEKA's XMEANS clusterer.

Each FC algorithm collected 20*k* pages. The timeout for the connection with any page and to read its content is 5 seconds, each. For a better performance, the crawler does not make other attempts to retrieve pages for which these timeouts expire.

### 5.1.2 Results

For each algorithm, with both sets of seed URLs, we have plotted in a graphic the evolution of the average precision as the number of collected pages increases. For comparing whether the result of one algorithm is better than the result of another, we compare the area under the curve (AUC) of the graphic. We use the sum of all average precisions to estimate the AUC of the algorithms.

Tables 5.4 and 5.5 show the AUC obtained in the FC experiments, for the software engineering and the dermatology domains, respectively. The leftmost columns show the algorithms mnemonics and the other ones represent the values of the AUC of the FC experiments, using, first, the set of bad seeds and then the set of good seeds. Results in both languages are detailed. The scores are grouped by the base algorithm and the variation that achieved the best result for each language and set of seed URLs is in bold.

In Figures 5.2 to 5.5, the average precisions over the number of collected pages for all the algorithms using both set of seed URLs and both languages are represented. For every base algorithm, we have chosen only the ones with the best results, which are in bold in Tables 5.4 and 5.5.

An analysis on these results allows us to observe a few things. We have established 12 different criteria for ranking the FC algorithms, based on a set of 3 parameters: (*i*) the set of seed URLs, (*ii*) the target domain and (*iii*) the language of the documents, or

Table 5.4: All the AUCs for the FC algorithms for the software engineering domain.

| Algorithm | AUC - Bad Seeds | | AUC - Good Seeds | | Average |
|---|---|---|---|---|---|
| | Portuguese | English | Portuguese | English | Ranking |
| UC | **80.49** | **175.62** | **405.00** | **950.76** | **11.33** |
| BFS-1G | 1035.11 | 3722.88 | 1645.69 | **7061.90** | **3.17** |
| BFS-2G | 303.81 | **4425.51** | 1476.67 | 4098.74 | 6.50 |
| BFS-3G | **1159.66** | 3576.64 | **1658.57** | 3865.98 | 5.83 |
| SS-1G | 188.5 | **5843.49** | **794.13** | 6875.90 | **4.67** |
| SS-2G | **225.24** | 5765.58 | 608.10 | 6881.34 | 5.33 |
| SS-3G | 192.99 | 3465.41 | 725.27 | **6941.03** | 7.00 |
| MWEBFS-2G | **1007.88** | **5636.19** | 1623.36 | 3824.29 | **5.33** |
| MWEBFS-3G | 416.36 | 3499.42 | **1642.70** | **3943.67** | 6.67 |
| MWESS-2G | 184.59 | 5506.60 | **801.82** | 6885.24 | 5.33 |
| MWESS-3G | **203.60** | **5703.13** | 628.84 | **6900.70** | **5.17** |
| HMM | **73.08** | **171.58** | **356.67** | **1324.53** | **11.67** |

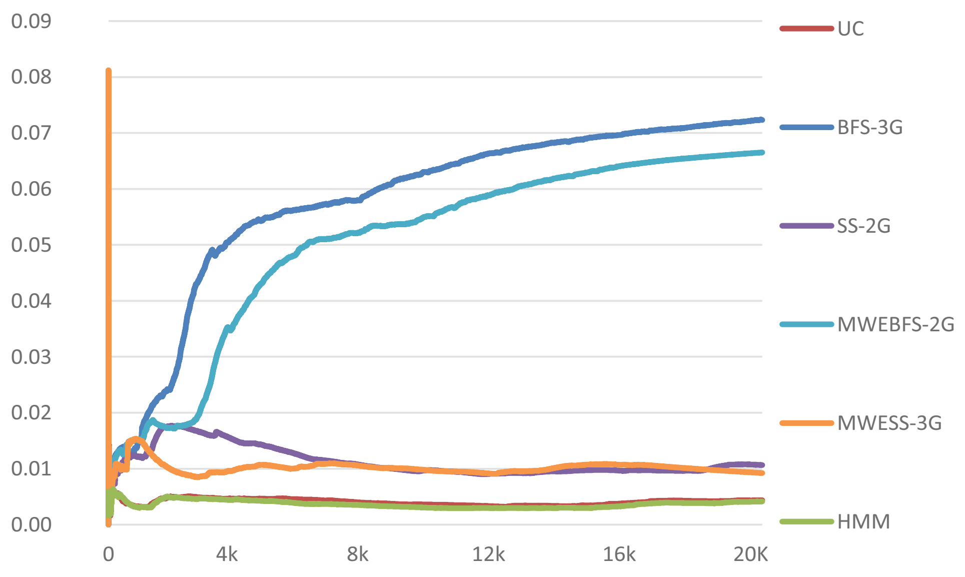
Table 5.5: All the AUCs for the FC algorithms for the dermatology domain.

| Algorithm | AUC - Bad Seeds | | AUC - Good Seeds | | Average |
|---|---|---|---|---|---|
| | Portuguese | English | Portuguese | English | Ranking |
| UC | **47.26** | **9.17** | **153.09** | **420.17** | **10.83** |
| BFS-1G | 158.22 | 1822.73 | **744.93** | 2645.03 | 2.83 |
| BFS-2G | 107.01 | 416.70 | 736.74 | 2657.95 | 5.33 |
| BFS-3G | **158.62** | **1840.36** | 736.24 | **2702.88** | **2** |
| SS-1G | 16.04 | 736.48 | 124.87 | 1180.98 | 9.83 |
| SS-2G | **85.53** | **1039.94** | **192.63** | **1194.44** | **6.17** |
| SS-3G | 35.14 | 1014.48 | 159.5 | 1193.83 | 7.83 |
| MWEBFS-2G | **504.25** | 475.84 | 480.26 | 2046.51 | 5.33 |
| MWEBFS-3G | 105.63 | **2127.81** | **755.02** | **2474.52** | **3** |
| MWESS-2G | 106.76 | 811.89 | **162.44** | 929.67 | 8.17 |
| MWESS-3G | **119.69** | **1041.05** | 145.56 | **1204.57** | **6.17** |
| HMM | **57.53** | **1.82** | **177.29** | **209.06** | **10.5** |

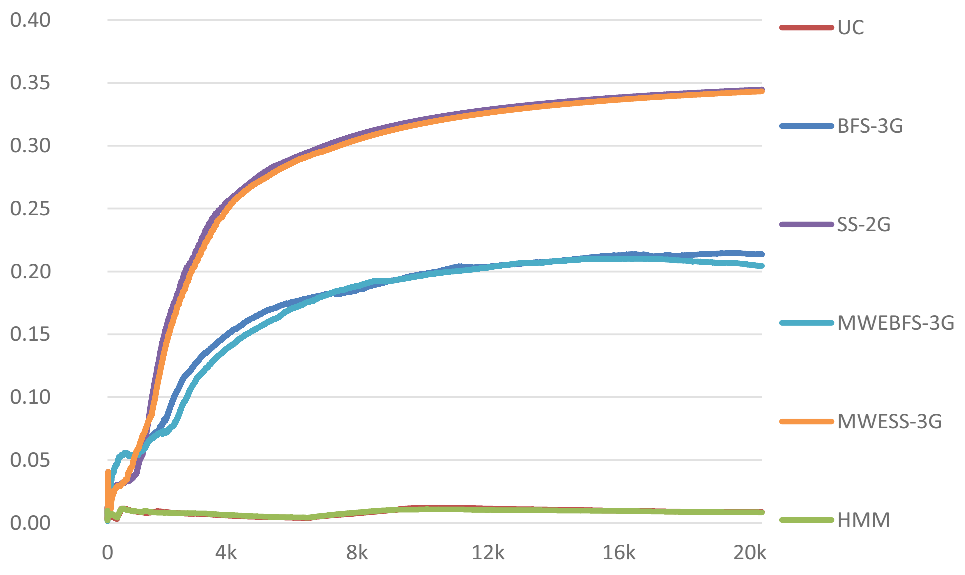
both languages (*i.e.*, the sum of the AUCs for English and Portuguese languages). To summarize the results, we consider the average ranking of each FC algorithm over the 6 possible variations of the first and the last of these parameters, maintaining only the target domain.

The $BFS-1G$ and the $BFS-3G$ obtained average rankings of 3 and 3.92, respectively. Although they presented the best results in some of the ranking criteria, they also showed bad results ($7^{th}$ to $9^{th}$ positions, out of 12) while using criteria like the software engineering domain, the set of bad seeds and the English language.

It is also important to investigate whether using the MWE based strategy, detailed in Section 4.2.2, is better than using the regular n-gram approach, explained in Section 4.2.1. This investigation can be done by comparing the n-gram based algorithms with their respective MWE-based variations — maintaining all the other parameters — on the average precision over the number of collected pages, for the quality of the FC. In either case, we analyzed when the result of one approach outperformed the other. The analysis, howe-



(a) Results for the Portuguese language.

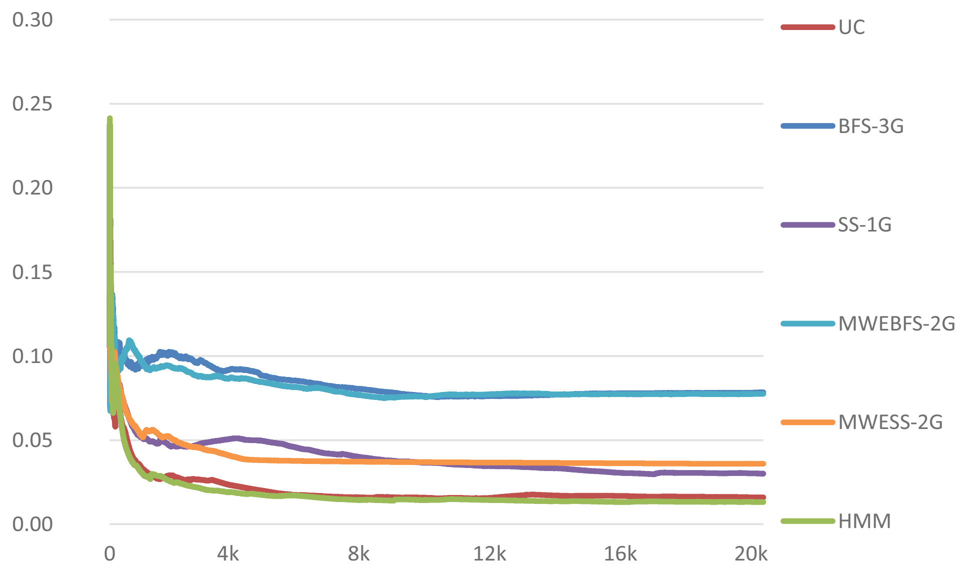


(b) Results for the English language.

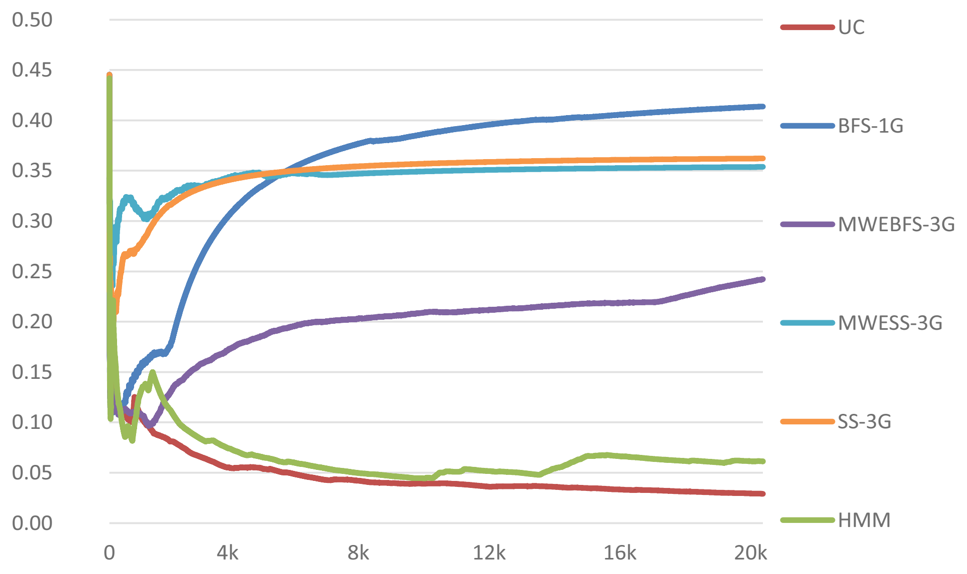
Figure 5.2: Average precision over the number of collected pages for the best variations of the base algorithms, for the software engineering domain, using the set of bad seeds.

ver, did not indicate that one approach is superior to the other, because none of them was significantly better too many times.

Although none of the algorithms can be said to be the most effective for FC we can affirm that *UC* and *HMM* noticeably present the worst performances. About the *HMM* algorithm, it is pertinent to observe that differently from the work of LIU; JANSSEN; MILIOS (2006), where the authors could count on the availability of specialists users



(a) Results for the Portuguese language.

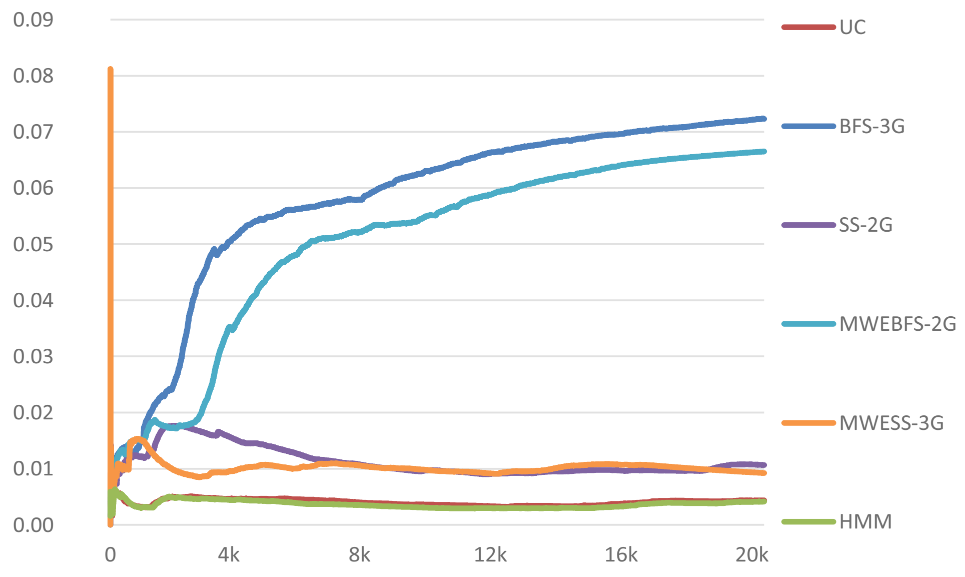


(b) Results for the English language.

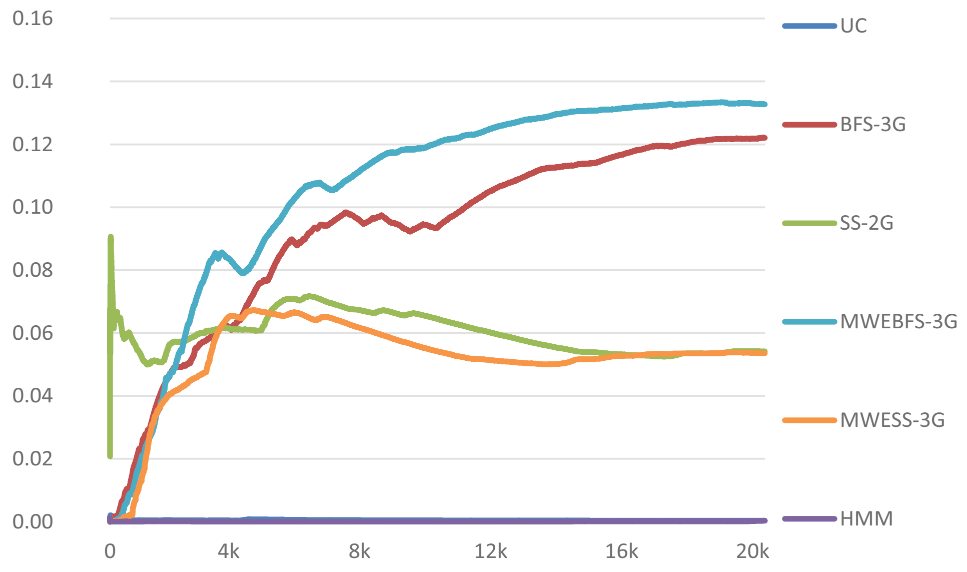
Figure 5.3: Average precision over the number of collected pages for the best variations of the base algorithms, for the software engineering domain, using the set of good seeds.

working on long browsing sessions for training the crawler, we had to manually perform some short browsing sessions. Hence, this could have led to the bad results presented by the *HMM* algorithm, in our experiments.

Analyzing the impact of varying the 3 ranking parameters lead us to the conclusion that they are also factors of high impact on the resulting AUC. With rare exceptions, the performance of the algorithm were significantly better when using the set of good seeds,



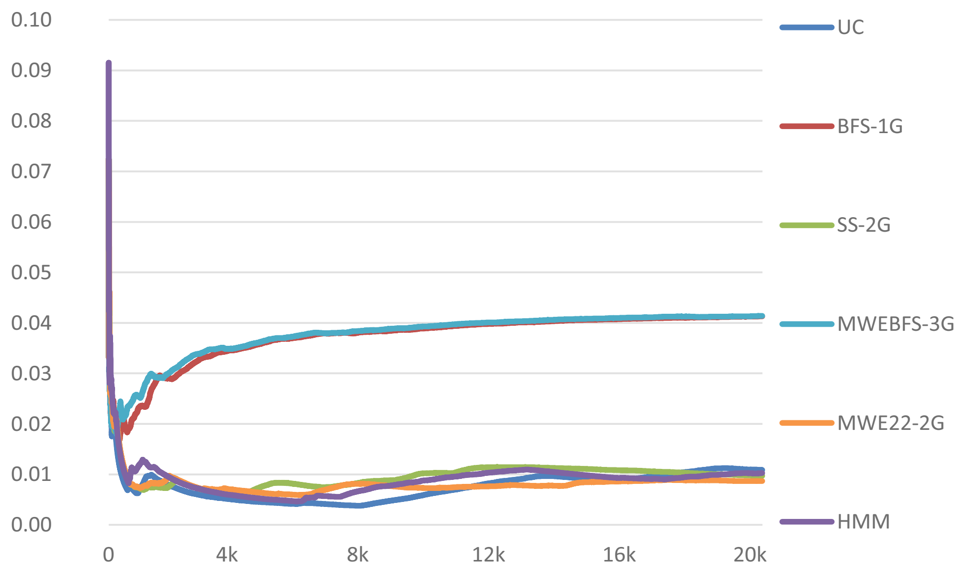
(a) Results for the Portuguese language.



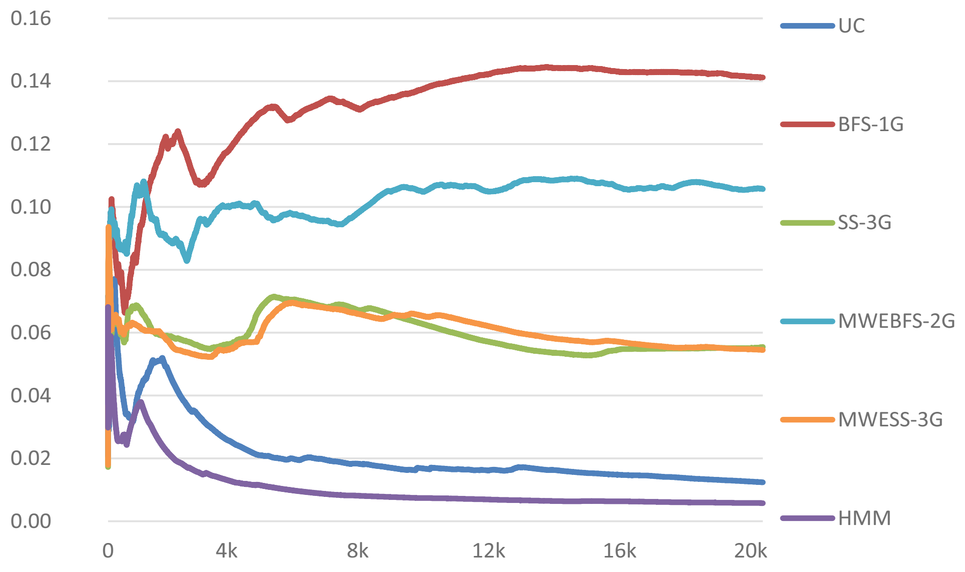
(b) Results for the English language.

Figure 5.4: Average precision over the number of collected pages for the best variations of the base algorithms, for the dermatology domain, using the set of bad seeds.

although the quality of the algorithms when using bad seeds in the English language were less susceptible to the effect of using the set of bad seeds, probably due to the larger availability of pages written in English on the web. The results from the FC in the software engineering domain were, in general, superior to the ones of dermatology, probably because the former is a domain with less domain-specific jargon than the later. Crawling for documents in English also appeared to be an easier task than collecting texts



(a) Results for the Portuguese language.



(b) Results for the English language.

Figure 5.5: Average precision over the number of collected pages for the best variations of the base algorithms, for the dermatology domain, using the set of good seeds.

in Portuguese. This may be due to a larger availability of English documents in the web. The larger availability of software engineering documents in the web can also explain that we obtained better results for this domain than for dermatology.

## 5.2 Domain-Specific Statistical Machine Translation

In this Section, we detail the linguistic resources and the tools used for every procedure involved in the SMT process, along with the results of these experiments.

### 5.2.1 Experimental Setup

Section 5.2.1.1 presents the details about the training of all the SMT algorithms and in 5.2.1.2, we explain how we have used and assessed them.

#### *5.2.1.1 SMT Training*

Prior to training the SMT algorithms, we needed to apply a set of preprocessing procedures, many of them performed with simple tools available in the UNIX terminal. First, we have concatenated the collected documents into a single one and converted them to the $UTF-8$ character encoding.

The output of these procedures were used as input to the sentence splitter script. This script was written in Python and uses the Natural Language ToolKit (NLTK) (LOPER; BIRD, 2002) to separate the sentences and write an output file with one sentence per line. Sentences with 23 or more words were discarded. This was done because we have noticed that most of them were cases where the NLTK's sentence splitter made some kind of mistake and concatenated multiple sentences together. The number 23 as a limit for the size of the sentence was not arbitrary. In the work of GRANADA et al. (2012), an analysis done in the corpora they have collected indicated that the average number of words in a sentence was not greater than 21 — 14.15 for English and 20.07 for Portuguese.

Then, the `sort` command was used for removing the duplicate sentences. Another Python script, using the LangID tool (LUI; BALDWIN, 2012), analyzed these sentences and filtered out the ones that it considered to be written in another language. The resulting file is tokenized by the Moses script for tokenization (KOEHN et al., 2007) .

The tokenized files were, then, used to build the LMs of order 5. The tool used for this task was the KenLM (HEAFIELD, 2011). First, it built *arpa* files, pruning n-grams of order 3 or higher that happened only once in the corpus, in order to reduce the size of the files. Then, all the generated files were binarized, for the SMT software to access its data faster.

The generation of the bilingual domain-specific lexicon was done with a program written in Java. It calculated the similarities between 1000 terms in the source language and 1000 in the target. Both sets of terms were extracted from the collected corpora by sorting all its words by the number of times they occur, times their IDFs in the generic collections mentioned above. The output of this program is a text file in the format of the phrase table used by the SMT software.

The MERT algorithm was used in order to search for the optimal weights for the features represented by both generic and domain-specific corpora. This algorithm needs a parallel corpus as development data for learning these weights. The parallel corpus we have used was the *Tortoise SVN User Manual*, written in English and Portuguese, because it was the one more related to the software engineering domain that we had access to. The quality of the translations after the training, however, decreased drastically. This problem probably occurred due to an overfitting on the development data, since it was related to the software engineering domain, but contains only the part of the jargon that is related to software versioning. Hence, we did not use the weights obtained with the MERT process.

For every focused crawling algorithm, we have built a set of 6 different SMT systems.

These variations were obtained by changing the values of two parameters. The first one is the variation of the set of seed URLs during the FC. The other parameter is the kind of domain-specific data that is used. This allows for three values: one for using only the domain-specific LM, one for using only the phrase table generated from the comparable corpora and one for combining both of them. Since there are 12 algorithms, including the variations of the base algorithms, we have a total of 72 SMT systems. By varying the parameters of the $BFS-1G$ algorithm, for example, we get the following 6 variations:

- $BFS-1G_{GOOD-LM}$: Corpora collected using the set of good seeds, using only the LM from the domain-specific corpus collected in the target language.
- $BFS-1G_{GOOD-PT}$: Corpora collected using the set of good seeds, using only the PT extracted from the domain-specific collected corpora.
- $BFS-1G_{GOOD-LMPT}$: Corpora collected using the set of good seeds, using both LM and PT from the domain-specific corpora.
- $BFS-1G_{BAD-LM}$: Corpora collected using the set of bad seeds, using only the LM from the domain-specific corpus collected in the target language.
- $BFS-1G_{BAD-PT}$: Corpora collected using the set of bad seeds, using only the PT extracted from the domain-specific collected corpora.
- $BFS-1G_{BAD-LMPT}$: Corpora collected using the set of bad seeds, using both LM and PT from the domain-specific corpora.

##### *5.2.1.2 Using and Assessing the SMT Systems*

The SMT software we chose to use is the Moses Toolkit (KOEHN et al., 2007). We have built a generic phrase table, with parallel data from the JRC[21] and the Europarl[22], in English and Portuguese, that was used in all SMT systems. Also, the Portuguese texts of both of these corpora were used to build a generic LM, also used by all systems. To construct each system, we only had to specify the data to be used by the Moses translator in a config file. *E.g.*, the file of the $BFS-1G_{GOOD-LMPT}$ system specifies that both domain-specific and generic LMs and phrase tables should be used.

To select the sentences that the SMT systems should translate, we used a method that did not bias towards any of them. First, we used the method of the work of GRANADA et al. (2012) and submitted a set of queries to the Yahoo! Search BOSS (Yahoo!, 2014). The resulting URLs were all stored, the documents pointed by them were retrieved and the same procedure applied to the corpora collected using the FC algorithms. Then, the sentences were shuffled and manually filtered, until we could acquire 100 valid, domain-specific, sentences. This procedure was applied for both dermatology and software engineering domains.

We also needed a gold-standard, for us to be able to compare the automatically generated translations with them. For the dermatology domain, we could gather 5 references. 4 professional translators translated all 100 sentences and the last reference was built collaboratively by 3 Linguistics students. For the domain of software engineering, we acquired 4 references, each of them translated by a specialist in the domain, including the authors.

[21]`http://optima.jrc.it/Acquis/index_2.2.html`
[22]`http://www.statmt.org/europarl/`

Instead of calculating a single BLEU score for each SMT system, using all the available references, we split each file generated by the SMT and each reference in 25 small parts of 4 sentences, to have a greater number of scores to compare and, thus, calculate whether they are significantly different or not, similarly to the pairwise comparison done in KOEHN (2010), that counts how many times one SMT system outperforms the other when comparing each sentence they have translated. Then, we computed the BLEU scores using every part of the automatically translated texts as a candidate translation and the corresponding part of a reference. Assessing an SMT system trained on the dermatology domain, for instance, generates 125 BLEU scores (25 parts $\times$ 5 references). Evaluating the systems trained on the software engineering, in turn, generates 100 BLEU scores (25 parts $\times$ 4 references).

Besides the SMT systems explained in Section 5.2.1.2, we also built two more types of systems. Below, we detail both of them, along with their respective mnemonics.

- ***BL:*** The baseline system, without the addition of any kind of domain-specific data to the generic SMT system. It does not allow for any of the *GOOD*/*BAD* or the *LM*/*PT*/*LMPT* variations, since they all refer to the domain-specific data.

- ***WP:*** An SMT system based on data from Wikipedia. For collecting the data, we developed a special kind of crawler, which starts from a page that describes a category on Wikipedia (*e.g.*, `http://en.wikipedia.org/wiki/Category:Software_engineering`), visits the articles that it contains, adds its subcategories in a queue and continues the process recursively. The LM is built using the texts of the articles that the starting category and its subcategories point to. To construct the phrase tables, we used an approach based on the cross-language links. We considered the title of a given article to be the translation of the title of the article pointed by its cross-language link. Hence, the titles of the articles found during this crawl and the title of the articles pointed by its corresponding cross-language links are used as training examples for building the phrase table. The *WP* SMT system allows the *LM*, the *PT* and the *LMPT* variations, because the constructed LMs and the phrase tables can be used or not.

### 5.2.2 Results

This Section presents the results of the SMT experiments. We represent them by the arithmetic mean of all the computed BLEU scores. The SMT systems that are significantly better than the *BL* system are indicated with the $^{\diamondsuit}$ signal. The ones significantly better than the corresponding variation (*LM*, *PT* or *LMPT*) of the *WP* system is indicated with the $^{\spadesuit}$ signal. We consider an SMT system to be significantly better than another when there is a 95% confidence level in Student's t-test (STUDENT, 1908). Tables 5.6 and 5.7 show, respectively, the results for the domains of software engineering and dermatology.

The line of the *BL* system contains the same value repeated 6 times, in order to allow comparison with the other systems. The results of the *WP* systems could appear only in 3 columns, because they do not vary with the *GOOD* or *BAD* parameters, but they are also replicated, in order to allow the comparison with the SMT systems generated with the data collected with the FC algorithms.

Just like in the analysis of the FC results, we have defined 12 different ranking criteria, according to a set of 3 parameters: (*i*) the set of seed URLs used to collect the corpora, (*ii*) the domain of interest and (*iii*) the strategy used for combining the domain-specific and the generic data for building the SMT system (*LM*, *PT* or *LMPT*). We observe that some

Table 5.6: All average BLEUs for the software engineering domain.

| Algorithm | BLEU - Bad Seeds | | | BLEU - Good Seeds | | | Average |
|---|---|---|---|---|---|---|---|
| | LM | PT | LMPT | LM | PT | LMPT | Ranking |
| BL | 14.73 | 14.73$^{\spadesuit}$ | 14.73$^{\spadesuit}$ | 14.73 | 14.73$^{\spadesuit}$ | 14.73$^{\spadesuit}$ | 4.5 |
| WP | 16.47$^{\diamondsuit}$ | 9.27 | 10.79 | 16.47$^{\diamondsuit}$ | 9.27 | 10.79 | **4** |
| UC | 15.34 | 8.11 | 6.63 | 16.59$^{\diamondsuit}$ | 5.9 | 5.44 | **10** |
| BFS-1G | 13.18 | 6.44 | 3.29 | 18.49$^{\diamondsuit\spadesuit}$ | 6.95 | 7.64 | 9.5 |
| BFS-2G | 12.29 | 8.09 | 4.52 | 18.68$^{\diamondsuit\spadesuit}$ | 6.45 | 7.86 | **8.17** |
| BFS-3G | 17.12$^{\diamondsuit}$ | 6.04 | 6.62 | 17.89$^{\diamondsuit\spadesuit}$ | 6.15 | 7.43 | 8.5 |
| SS-1G | 15.22 | 7.2 | 6.78 | 17.68$^{\diamondsuit\spadesuit}$ | 7.71 | 7.82 | **7.17** |
| SS-2G | 16.36$^{\diamondsuit}$ | 7.42 | 6.26 | 17.47$^{\diamondsuit\spadesuit}$ | 7.74 | 7.22 | 7.5 |
| SS-3G | 15.86 | 7.27 | 5.58 | 16.76$^{\diamondsuit}$ | 7.64 | 7.28 | 8.67 |
| MWEBFS-2G | 17.83$^{\diamondsuit\spadesuit}$ | 8.25 | 10 | 17.86$^{\diamondsuit\spadesuit}$ | 5.99 | 7.89 | **5.17** |
| MWEBFS-3G | 11.84 | 6.58 | 2.47 | 18.43$^{\diamondsuit\spadesuit}$ | 6.84 | 7.98 | 9.5 |
| MWESS-2G | 14.22$^{\diamondsuit}$ | 7.11 | 6.81 | 16.9$^{\diamondsuit}$ | 8.03 | 5.96 | 8.83 |
| MWESS-3G | 14.67$^{\diamondsuit\spadesuit}$ | 7.76 | 6.87 | 18.72$^{\diamondsuit\spadesuit}$ | 7.28 | 7.63 | **6.33** |
| HMM | 16.81 | 8.94 | 8.43 | 15.47$^{\diamondsuit}$ | 7.98 | 7.21 | **7.17** |

of the algorithms were assigned better average rankings. The 4 algorithms with the best average rankings, $SS-2G$, $MWEBFS-2G$, $MWESS-2G$ and $BFS-2G$, in that order, use bigrams as the textual unit. However, just like observed when analyzing the AUC of the FC algorithms, all of these 4 are also low ranked, reaching the 11th position, out of 12, when using a sorting criterion like set of bad seeds, the software engineering domain and the *LM* strategy.

The use of the strategies for FC based on MWEs can also be compared to the ones based on n-grams by the quality of the resulting SMT systems. We analyzed how many times one algorithm was significantly superior to the other, with a confidence level of 95%, in the Student's t-test. But similarly to the FC results, none of the approaches proved to be superior to the other.

Even with a shallow analysis we can observe the difference on the BLEU scores caused by changing the approach for combining generic and domain-specific data for SMT. The results in Tables 5.6 and 5.7 clearly show that the *LM* strategy, which only uses the in-domain LMs, is better than the others, because all the presented average BLEU scores are superior to those obtained with any of the other two variants.

This points out another question, about the quality of the generated phrase tables, given their clearly negative effect on the quality of the translations. Hence, we performed an examination on the generated phrase tables. The terms selected for alignment are, indeed, from the topic of interest. However, the problem is evident, due to the difficulty of finding valid pairs of semantic related terms. For instance, the terms "lymphangioma" and "vitiligo" are from the dermatology domain, but they are not supposed to be considered related, as indicated in the phrase table.

The quality of the *HMM* algorithm, just like for FC, was very low. Again, we attribute this to the difficulty of properly training the algorithm.

Table 5.7: All average BLEUs for the dermatology domain.

| Algorithm | BLEU - Bad Seeds | | | BLEU - Good Seeds | | | Average |
|---|---|---|---|---|---|---|---|
| | LM | PT | LMPT | LM | PT | LMPT | Ranking |
| BL | 7.71 | 7.71$^{\spadesuit}$ | 7.71 | 7.71 | 7.71$^{\spadesuit}$ | 7.71 | 5.5 |
| WP | 8.82$^{\diamondsuit}$ | 4.57 | 6.95 | 8.82$^{\diamondsuit}$ | 4.57 | 6.95 | **5.17** |
| UC | 9.49$^{\diamondsuit}$ | 4.26 | 7 | 11.26$^{\diamondsuit\spadesuit}$ | 3.82 | 7.69 | **4.83** |
| BFS-1G | 7.8 | 3.78 | 3.45 | 11.27$^{\diamondsuit\spadesuit}$ | 3.4 | 5.36 | 11.17 |
| BFS-2G | 7.97 | 3.83 | 4.43 | 11.46$^{\diamondsuit\spadesuit}$ | 4.18 | 8.07$^{\spadesuit}$ | **6** |
| BFS-3G | 8.05 | 3.64 | 3.61 | 11.16$^{\diamondsuit\spadesuit}$ | 3.58 | 5.48 | 10.83 |
| SS-1G | 8.2 | 3.69 | 4.09 | 11.13$^{\diamondsuit\spadesuit}$ | 3.89 | 6.39 | 9.17 |
| SS-2G | 10.44$^{\diamondsuit\spadesuit}$ | 4.08 | 5.81 | 11.02$^{\diamondsuit\spadesuit}$ | 4.52 | 7.85 | **4.83** |
| SS-3G | 8.45 | 3.89 | 4.54 | 11.7$^{\diamondsuit\spadesuit}$ | 3.83 | 7.04 | 6 |
| MWEBFS-2G | 8.16 | 4.56 | 6.23 | 11.01$^{\diamondsuit\spadesuit}$ | 3.69 | 6.69 | **7.67** |
| MWEBFS-3G | 8.14 | 3.53 | 3.77 | 10.65$^{\diamondsuit\spadesuit}$ | 3.24 | 5.23 | 12.33 |
| MWESS-2G | 10.06$^{\diamondsuit\spadesuit}$ | 4.16 | 6.23 | 11.34$^{\diamondsuit\spadesuit}$ | 4 | 6.84 | **4.83** |
| MWESS-3G | 7.18 | 4.29 | 4.14 | 11.11$^{\diamondsuit\spadesuit}$ | 3.94 | 6.57 | 8.83 |
| HMM | 8.33 | 3.4 | 4.37 | 11.91$^{\diamondsuit\spadesuit}$ | 3.57 | 7.43 | **7.83** |

## 5.3 Discussion

After the evaluation of the FC and the SMT strategies, there are still some other questions that deserve a careful analysis. Is there a correlation between the quality of the FC algorithms, considering their AUC, and the BLEU score of the SMT systems generated from the data collected by them? Is the FC a good approach for overcoming the problem of scarceness of domain-specific parallel corpora available on the web? Which FC algorithm should be considered the default one, for non expert users willing to acquire comparable corpora with our FC system?

To answer the first question, we have made our analyses separately in each domain and considered that algorithms with different sets of seed URLs are distinct (*e.g.*, $BFS-1G_{GOOD}$ and $BFS-1G_{BAD}$ are considered distinct algorithms), because it makes sense. Since we concluded that the best approach for combining in-domain and out-of-domain data is the *LM* approach, we sought for a correlation between the SMT quality using this strategy and the quality of the FC only in Portuguese, because the LM is built only using the target language documents.

The value of the average BLEU score is a real number that ranges from 0, corresponding to the worst translations, to 100 (or 1), for exact translations, while the value of the AUC of the FC algorithms is a real number that ranges from 0 to 20$k$, which is the number of collected pages. Hence, to make both of them range between the same values, we assigned, for each algorithm, its position in the ranking of 24 algorithms (12 algorithms $\times 2$ sets of seed URLs), making them to be a natural number that varies between 1 and 24. Then we calculated Kendall's tau ranking correlation (KENDALL; GIBBONS, 1990) between the quality rankings of both processes. For the domain of software engineering, the correlation value was 0.646, with $p-value = 9 \times 10^{-5}$ and for dermatology, the obtained value for the correlation was 0.319, with $p-value = 0.03$. Also, the confirmation of this correlation suggests that the standard metrics for the FC are good indicatives on the quality on the SMT systems that can be built with its collected data.

Although we cannot affirm that one FC algorithm presents better results, either considering the quality of the FC or the SMT, it is important for some users to know which algorithm to use, when they are not aware of the details of their methods. Besides the quality, we should also consider that different algorithms have different computational costs for running. The ones that use higher order n-grams as textual units need to store a considerably larger amount of information, from n-gram IDF and PMI values, in memory. Thus, it may be a fair decision to advise users to use the $BFS-1G$ or the $SS-1G$, that use only unigrams and, therefore, do not depend on the availability of much computational resources. The $BFS-1G$ presented, in general, better results for FC and the $SS-1G$ performed better for the domain adaptation task.

The bad results obtained for both FC and SMT experiments for the dermatology domain, specially when compared to the results of software engineering, may be due to the dermatology domain being less similar to a generic corpus than the software engineering domain. In other words, performing those tasks with the dermatology topic is harder because its jargon is composed of words that occur less times in a generic corpus, like a collection of newspaper texts.

The large number of times that the SMT systems using the collected data significantly overcame the performance obtained by the $BL$ and $WP$ seems to be enough for saying that the FC is a good alternative for overcoming the obstacle of not having many parallel resources for some specific domains and some specific languages available on the web. Nonetheless, since these results were obtained almost exclusively with the *LM* approach, which only explore the LMs, if we manage to build better domain-specific phrase tables from the comparable corpora, it can be an even better method for this.

# 6 CONCLUSIONS

SMT is a process that relies on large amounts of parallel texts written on both source and target languages to be able to work properly. This kind of resource, however, is not easy to find, for many specific domains or some under-resourced languages. When parallel corpora are not available, the use of comparable corpora, *i.e.*, texts that are from the same topic, is an alternative.

In the work presented in this dissertation, we have proposed the use of FC as a means of acquiring comparable corpora. For verifying if this is a valid idea, we have applied several FC algorithms for collecting comparable corpora and used them as domain-specific data for performing domain adaptation for SMT.

Besides experimenting some FC algorithms already present in the literature, we have also proposed new ones. Some of them are based on the modification of the textual unit from single words to n-grams. Others try to explore the expressive power of the MWEs, by the addition of the MWE Factor in the classic TF-IDF model. For the SMT domain adaptation task, the collected corpora can be used for generating LMs and phrase tables, by trying to align terms to its translation in the other language.

We conducted experiments of FC and SMT on the software engineering and the dermatology domains and on the English and Portuguese languages, varying the quality of the set of seed URLs. The analyses done on the obtained results showed that FC may be a valid alternative for SMT domain adaptation, when there is no parallel corpora for a given topic or language. Also, the correlation found between the quality of the FC and of the SMT, using the standard metrics, suggests that the quality of the former may be a good indicative on how good the later is.

As a result of this work the following resources and tools were developed and will be made available to the research community:

- An easily extendable FC, written in Java, with a several algorithms already implemented.
- Translation gold-standards in the domains of dermatology and software engineering, for English and Portuguese.
- A collection of IDF and PMI values for words and n-grams, calculated from generic texts in English and Portuguese.

We can identify some parts of this strategy that allow room for improvement. For future work we envisage extending the MWE Factor, used for the $MWEBFS-2G$, the $MWEBFS-3G$, the $MWESS-2G$ and the $MWESS-3G$ algorithms. Using the PMI of the n-gram as the only indicative of whether it is a MWE is a naive strategy. Instead, we could gather a number of association metrics for a set of n-grams, like the *Dice's coefficient*, the *t-score* or the *log likelihood*, manually annotate them as MWEs or not and calculate a linear regression for estimating how likely a novel, unseen, n-gram is to be considered a MWE. We think that this is a promising idea, because even when simply using the PMI as the only source for building the MWE Factor, the MWE based approaches obtained results similar to the n-gram based ones.

To better validate the proposed ideas, a greater number of FC algorithms, of domains and languages, varying the directions of the translations (translate from English to Portuguese and vice-versa), could be done. Using some adaptive FC algorithms, the genomics domain and the French language, for example, should help doing this. Also, it would be interesting to perform analyses using other metrics for assessing the quality of both processes. For FC, the harvest rate would be an alternative, although it has the inconvenience of classifying pages as related to the topic of interest or not, not assigning them a continuous score for estimating how much they are related. The evaluation of SMT is a more complex task, thus, allowing a wider range of possible metrics to be used. Some alternatives to the BLEU score are TER (SNOVER et al., 2006) or METEOR (BANERJEE; LAVIE, 2005). Also, the work of GIMÉNEZ; AMIGÓ (2006) could be employed or some manual evaluation, where linguists or domain specialists observes 2 or more automatic translations for a given sentence and decide which is the best one.

Further and thorough examinations of the causes of the low quality of the alignment between the words quality seems to be mandatory. Enhancing the quality of this process would probably greatly raise the SMT quality.

In all the approaches for combining in-domain and out-of-domain data, *LM*, *PT* and *LMPT*, the weights of the features represented by both should be optimized. We must seek for some weighting strategy that maximizes the quality of the translation, not falling in the overfitting problem, found when we have tried to use the MERT algorithm with the *Tortoise SVN User Manual* as the development data.

With all these improvements properly investigated and, if possible, developed, we could adapt the FC system and its respective text processing procedures to be available as a user friendly extension for performing domain adaptation in the Moses SMT toolkit (KOEHN et al., 2007).

The realization of this work, resulted in the following papers:

- LARANJEIRA, B.R.; MOREIRA, V.P.; VILLAVICENCIO, A. (2013) **Correlation between the quality of focused crawlers and the linguistic resources obtained from them**, In: 12$^o$ WORKSHOP DE TESES E DISSERTAÇÕES EM BANCOS DE DADOS, WTDBD, 28$^o$ SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, SBBD, Recife, Brazil: This paper presented the main idea of the work of this dissertation, explaining the correlation between the quality of the FC and of the SMT algorithms that we expected to find.

- LARANJEIRA, B.R.; MOREIRA, V.P.; VILLAVICENCIO, A.; RAMISCH, C.; FINATTO, M.J. (2014) **Comparing the Quality of Focused Crawlers and of the Translation Resources Obtained from them**, In: Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC), Reykjavik, Iceland. **Qualis A2**. In this paper we have tried to establish a correlation between the quality of the FC algorithms and the quality of the SMT systems using the data they have collected for building in-domain LMs. The expected correlation, however, was not confirmed.

Currently, we are working on a paper submission for the **Computer Speech and Language Journal (Qualis B1)**. The paper will include all the progress presented in this dissertation along with the next studies and experiments on the process of finding equivalent terms between both source and target languages, which we expect to be one of the major novelties with respect to the last published work.

# REFERENCES

AIJMER, K.; ALTENBERG, B.; JOHANSSON, M. **Languages in contrast**: papers from a symposium on text-based cross-linguistic studies, lund, 4-5 march 1994. [S.l.]: Lund Univ. Press, 1996. v.88.

AKBARI TORKESTANI, J. An adaptive focused web crawling algorithm based on learning automata. **Applied Intelligence**, [S.l.], p.1–16, 2012.

BAEZA-YATES, R.; RIBEIRO-NETO, B. **Modern Information Retrieval**: the concepts and technology behind search (2nd edition) (acm press books). [S.l.]: Addison-Wesley Professional, 2011.

BANERJEE, S.; LAVIE, A. METEOR: an automatic metric for mt evaluation with improved correlation with human judgments. In: ACL WORKSHOP ON INTRINSIC AND EXTRINSIC EVALUATION MEASURES FOR MACHINE TRANSLATION AND/OR SUMMARIZATION. **Proceedings...** [S.l.: s.n.], 2005. p.65–72.

BANERJEE, S.; PEDERSEN, T. The Design, Implementation, and Use of the Ngram Statistic Package. In: FOURTH INTERNATIONAL CONFERENCE ON INTELLIGENT TEXT PROCESSING AND COMPUTATIONAL LINGUISTICS, Mexico City. **Proceedings...** [S.l.: s.n.], 2003. p.370–381.

BERTOLDI, N.; FEDERICO, M. Domain adaptation for statistical machine translation with monolingual resources. In: Proceedings of the Fourth Workshop on Statistical Machine Translation. **Anais...** [S.l.: s.n.], 2009. p.182–189.

BRODER, A. Z. et al. Syntactic clustering of the web. **Computer Networks and ISDN Systems**, [S.l.], v.29, n.8, p.1157–1166, 1997.

CHAKRABARTI, S. **Mining the Web**: discovering knowledge from hypertext data. [S.l.]: Morgan Kaufmann, 2002.

CHAKRABARTI, S.; BERG, M. Van den; DOM, B. Focused crawling: a new approach to topic-specific web resource discovery. **Computer Networks**, [S.l.], v.31, n.11, p.1623–1640, 1999.

CLAVEAU, V.; KIJAK, E.; FERRET, O. Improving distributional thesauri by exploring the graph of neighbors. In: INTERNATIONAL CONFERENCE ON COMPUTATIONAL LINGUISTICS, COLING 2014. **Anais...** [S.l.: s.n.], 2014. p.12–p.

DE BRA, P. et al. Information retrieval in distributed hypertexts. In: RIAO CONFERENCE, 4. **Proceedings...** [S.l.: s.n.], 1994. p.481–491.

FELLBAUM, C. **WordNet**. [S.l.]: Wiley Online Library, 1998.

FRANCOIS, J.-M. **JAHMM**: an implementation of hidden markov models in java. Last accessed: February 8, 2015, `https://code.google.com/p/jahmm/`.

GIMÉNEZ, J. ús; AMIGÓ, E. IQMT: a framework for automatic machine translation evaluation. In: LREC. **Proceedings. . .** [S.l.: s.n.], 2006. p.685–690.

GRANADA, R. et al. A comparable corpus based on aligned multilingual ontologies. In: FIRST WORKSHOP ON MULTILINGUAL MODELING. **Proceedings. . .** [S.l.: s.n.], 2012. p.25–31.

GROSSMAN, D. A.; FRIEDER, O. **Information Retrieval**: algorithms and heuristics (the information retrieval series)(2nd edition). [S.l.]: Springer, 2004.

HALL, M. et al. The WEKA data mining software: an update. **ACM SIGKDD explorations newsletter**, [S.l.], v.11, n.1, p.10–18, 2009.

HEAFIELD, K. KenLM: faster and smaller language model queries. In: SIXTH WORKSHOP ON STATISTICAL MACHINE TRANSLATION. **Proceedings. . .** [S.l.: s.n.], 2011. p.187–197.

HERSOVICI, M. et al. The shark-search algorithm. An application: tailored web site mapping. **Computer Networks and ISDN Systems**, [S.l.], v.30, n.1, p.317–326, 1998.

HSU, C.; WU, F. Topic-specific crawling on the web with the measurements of the relevancy context graph. **Information Systems**, [S.l.], v.31, n.4, p.232–246, 2006.

IRVINE, A. et al. Measuring Machine Translation Errors in New Domains. **Transactions of the Association for Computational Linguistics**, [S.l.], v.1, p.429–440, 2013.

JHA, P. et al. Robots Exclusion Protocol. **Internation journal of emerging science and engineering**, [S.l.], v.2, n.5, 2014.

JURAFSKY, D.; MARTIN, J. H. **Speech and Language Processing (2nd Edition)**. [S.l.]: Pearson Prentice Hall, 2008.

KENDALL, M.; GIBBONS, J. D. **Rank Correlation Methods (Charles Griffin Book Series)**. [S.l.]: Oxford University Press, 1990.

KESKUSTALO, H.; HEDLUND, T.; AIRIO, E. UTACLIR-: general query translation framework for several language pairs. In: ACM SIGIR CONFERENCE ON RESEARCH AND DEVELOPMENT IN INFORMATION RETRIEVAL, 25. **Proceedings. . .** [S.l.: s.n.], 2002. p.448–448.

KOEHN, P. **Statistical Machine Translation**. [S.l.]: Cambridge University Press, 2010.

KOEHN, P. et al. Moses: open source toolkit for statistical machine translation. In: ANNUAL MEETING OF THE ACL ON INTERACTIVE POSTER AND DEMONSTRATION SESSIONS, 45. **Proceedings. . .** [S.l.: s.n.], 2007. p.177–180.

LÄUBLI, S. et al. Combining Statistical Machine Translation and Translation Memories with Domain Adaptation. In: NORDIC CONFERENCE OFCOMPUTATIONAL LINGUISTICS, 19. **Proceedings. . .** [S.l.: s.n.], 2013. p.331–341.

LEE, S.; KIM, S.; HONG, S. On URL normalization. **Computational Science and Its Applications–ICCSA 2005**, [S.l.], p.122–130, 2005.

LIN, D. Automatic retrieval and clustering of similar words. In: COMPUTATIONAL LINGUISTICS-VOLUME 2, 17. **Proceedings...** [S.l.: s.n.], 1998. p.768–774.

LIU, B. **Web Data Mining**: exploring hyperlinks, contents, and usage data (data-centric systems and applications). [S.l.]: Springer, 2011.

LIU, H.; JANSSEN, J.; MILIOS, E. Using HMM to learn user browsing patterns for focused web crawling. **Data & Knowledge Engineering**, [S.l.], v.59, n.2, p.270–291, 2006.

LOPER, E.; BIRD, S. NLTK: the natural language toolkit. In: ACL-02 WORKSHOP ON EFFECTIVE TOOLS AND METHODOLOGIES FOR TEACHING NATURAL LANGUAGE PROCESSING AND COMPUTATIONAL LINGUISTICS-VOLUME 1. **Proceedings...** [S.l.: s.n.], 2002. p.63–70.

LUI, M.; BALDWIN, T. langid. py: an off-the-shelf language identification tool. In: ACL 2012 SYSTEM DEMONSTRATIONS. **Proceedings...** [S.l.: s.n.], 2012. p.25–30.

MANNING, C. D.; RAGHAVAN, P.; SCHÜTZE, H. **Introduction to Information Retrieval**. [S.l.]: Cambridge University Press, 2008.

MCENERY, A.; XIAO, Z. Parallel and comparable corpora: what is happening. **Incorporating Corpora. The Linguist and the Translator**, [S.l.], p.18–31, 2007.

MCENERY, T.; WILSON, A. **Corpus Linguistics (Edinburgh Textbooks in Empirical Linguistics)**. [S.l.]: Edinburgh University Press, 2001.

MOHAMMAD, S. et al. Cross-Lingual Distributional Profiles of Concepts for Measuring Semantic Distance. In: JOINT CONFERENCE ON EMPIRICAL METHODS IN NATURAL LANGUAGE PROCESSING AND COMPUTATIONAL NATURAL LANGUAGE LEARNING (EMNLP-CONLL). **Proceedings...** [S.l.: s.n.], 2007. p.571–580.

Oswald, Derrick. **HTML Parser**. Last accessed: February 8, 2015, `http://htmlparser.sourceforge.net/`.

PADRÓ, M. et al. Comparing similarity measures for distributional thesauri. In: NINTH INTERNATIONAL CONFERENCE ON LANGUAGE RESOURCES AND EVALUATION (LRECŠ14), REYKJAVIK, ICELAND: EUROPEAN LANGUAGE RESOURCES ASSOCIATION (ELRA). **Proceedings...** [S.l.: s.n.], 2014.

PAPINENI, K. et al. BLEU: a method for automatic evaluation of machine translation. In: OF THE 40TH ANNUAL MEETING ON ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings...** [S.l.: s.n.], 2002. p.311–318.

PELLEG, D.; MOORE, A. et al. X-means: extending k-means with efficient estimation of the number of clusters. In: OF THE SEVENTEENTH INTERNATIONAL CONFERENCE ON MACHINE LEARNING. **Proceedings...** [S.l.: s.n.], 2000. v.1, p.727–734.

PIRKOLA, A. et al. FITE-TRT: a high quality translation technique for oov words. In: ACM SYMPOSIUM ON APPLIED COMPUTING, 2006. **Proceedings...** [S.l.: s.n.], 2006. p.1043–1049.

PORTER, M. F. Snowball: a language for stemming algorithms, 2001. **URL http://snowball. tartarus. org/texts/introduction. html**, [S.l.], 2009.

RAGHAVAN, S.; GARCIA-MOLINA, H. Crawling the Hidden Web. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES (VLDB 2001), 27. **Anais...** [S.l.: s.n.], 2001.

SAG, I. A. et al. Multiword expressions: a pain in the neck for nlp. In: **Computational Linguistics and Intelligent Text Processing**. [S.l.]: Springer, 2002. p.1–15.

SKADIŅA, I. et al. Collection of comparable corpora for under-resourced languages. In: FOURTH INTERNATIONAL CONFERENCE BALTIC HLT 2010. **Proceedings...** [S.l.: s.n.], 2010. p.161–168.

SNOVER, M. et al. A study of translation edit rate with targeted human annotation. In: AMERICAS. **Proceedings...** [S.l.: s.n.], 2006. p.223–231.

STUDENT. The probable error of a mean. **Biometrika**, [S.l.], p.1–25, 1908.

TALVENSAARI, T. et al. Focused web crawling in the acquisition of comparable corpora. **Information Retrieval**, [S.l.], v.11, n.5, p.427–445, 2008.

USZKOREIT, J. et al. Large scale parallel document mining for machine translation. In: INTERNATIONAL CONFERENCE ON COMPUTATIONAL LINGUISTICS, 23. **Proceedings...** [S.l.: s.n.], 2010. p.1101–1109.

Yahoo! **Yahoo! Directory**. Last accessed: December 30, 2013, `http://dir.yahoo.com/`.

Yahoo! **Yahoo! Search BOSS**. Last accessed: February 10, 2015, `https://developer.yahoo.com/boss/search/`.