Focused Crawling: algorithm survey and new approaches with a manual analysis

Ignacio García Dorado

Department of Electrical and Information Technology
Lund University

Advisor: Anders Ardö

April 7, 2008

Printed in Sweden
E-huset, Lund, 2008

Abstract

The exponential growth of the Internet makes the universal search engines try to address the scalability limitations with vast amounts of hardware and network resources and by distributing the crawling process across users, queries, or even client computers. Furthermore it makes increasingly difficult to discover topic relevant information that can be used in specialized portals and on-line search. To tackle this issue the focused web crawlers are emerging. They are a kind of crawlers that dynamically browse the Internet by choosing the most promising links in order to try to maximize the relevancy of the retrieved pages and thus saving significantly bandwidth and time. In this master thesis, an algorithm survey is done to find the best alternatives in the literature. With the best options (Graphic Context, Best-First and History Path) and two new approaches (proposed in this paper) an automatic and manual analysis is performed with two different document topic classifier algorithms: SVM and String-Matching. It will be shown that any of the alternatives in the literature outperforms the Best-First algorithm and just the two new approaches do it. It is shown that using both classifiers to define what it is 'best' in the Best-First algorithm obtains the best precision and accuracy. Moreover it is shown that the anchor text gives really valuable information than can be used to increase the precision through a learning process.

Table of Contents

Chapter 1

Introduction

Nowadays the Internet is becoming one of the main sources of information search. Young people use the search engines to find any kind and, actually, they will become the called 'Google generation' [1]. Because of this, the importance of search engines is increasing as well as the necessity of finding more accurate information.

On one hand the amount of electronic information is getting vast dimensions and the major web crawlers are trying to keep growing with distributed system, but even if it were feasible, it would be useless. The diversity of content in a general search makes that general queries obtain quite good results but when a really specific query is done, the results becomes irrelevant. On the other hand, there are users who want the Internet for their specific area, for those people a topic-specific search engine would be desirable. A way to address it would be to crawl the whole Internet and try to classify everything but the fast changes that some areas need would not be sufficient.

To create a web search engine, it needs documents to be indexed, therefore a web Crawler is used to fetch the pages. A crawler is a program that given some web pages (seeds) extracts the out-links and adds them to a list (that it is called frontier), when they are processed, it takes one from the list of web-pages and the process is repeated recursively. With few seeds it is able to find thousands of pages. This is possible because the Internet is like a gigantic directional-connected graph where the nodes are the pages and the joins are the links. It is directional because a web-link is from a page to just another. It depends on how the next link is taken from that list it will be fetched quite different subsets of the Internet first.

Therefore a new idea comes up, a Focused Crawler. This kind of crawlers try to find high-quality information on a specific subject as soon as possible and try to avoid irrelevant pages in order to the results would be as accurate as possible. Thus, a focused crawler is a program which fetches as much relevant pages as possible. It uses the link structure but the order of that list is important. This is because in this case it is not desirable to retrieve everything but to fetch high-quality pages. Thus, there are different algorithms or schedulers that try to predict which should be the next page to be crawled. The main difference between those algorithms is the kind of information that is used.

This is a good approach for three reasons:

1. The amount of web-pages is increasing constantly: the size of the visible Web is estimated to be greater than 29.7 billion pages on end of February 2007 [2]. Then it becomes to be a retrieval problem because the Web does not have just to be fetched once, it is really frequently in change. Furthermore and despite the fact that the major web crawlers try to grow their systems enough to index the whole Web, it is only a matter of time that it will become an impossible task.

2. At the same time that the search engines try to fetch as many pages as possible to follow the Internet expansion, it makes that a general query just obtain few good result which are mixed with plenty irrelevant pages.

3. The rising necessity of topic-search engines. The Internet has become one of the biggest and influential sources of information and since many users connect to the Web to find information from their area, they would use topic search engines. For instance, in the case of digital libraries, the collecting topic-specific information can be performed much faster if 'smart' strategies of crawling are applied.

.

A focused crawler does its task starting from a few relevant pages (seeds) and then follows promising links first to find the more relevant (topic related pages) as soon as possible. Thus it is needed a link-scheduler or algorithm that decides which will be the next link to follow. Since there are many ways to schedule and algorithms, there are many different focused crawlers. Therefore my task here is to:

- Study the state of the art and select the most promising algorithms.
- Implement them in a real crawler.
- Create an automatic and a manual comparison.
- Try to find out the best option.

Summarizing, the task in this master thesis is to find out which is the best algorithm to do a good Focused Crawler. In the literature can be found different approaches, it will be necessary to dig into it and find the best one.

The outline of the report is: In the chapter 2, the state of the art is presented. In chapter 4 is detailed what a topic definition is, how a topic can be described to be used in a Focused Crawler. The chapter 5 describes the architecture of the system where the testing has been done. Chapter 6 shows the algorithm alternatives that have been discarded and have been selected. In the chapter 7 it is explained how the text is scored, two document topic classifier algorithms will be used: SVM and String-Matching. In chapter 8 a general view of the further analysis is explained, the automatic analysis is presented in the chapter 9, and in the chapter 10 the manual analysis. In the chapter 11 other topic is used to compare the previous results and to check if they are easily extending. Finally in the chapter 12 the conclusions and ongoing research are exposed.

Chapter 2

State of the art

2.1 Introduction

Focused Crawlers can be used for different purposes that depend on which kind of information must be found. For each kind, there are some investigation that tries to find more accurate relevant pages in that narrow case. This makes that most researches cannot be extrapolated for a general use.

On the other hand it is necessary to distinguish between a Focused Crawler and just an efficient Crawler even though they are related. An example of efficient Crawling can be found in [3] that describes how PageRank[1] is an excellent ordering metric but this is just useful if we are just looking for hot pages but without a specific-topic (an ordering metric is used to sort the queue and obtain better results). We will not consider this as Focused Crawler.

2.2 Digital Libraries

A Digital Library is an electronic tool where are stored collections of information in digital formats.

There are many papers about this situation but the problem is that they are implemented for actually specific areas.

In [4] it is showed that it is possible to look onto the homepage of an author to find its missing papers. It works but it is too specific because it uses the structure of the University web-pages and the researchers pages. And that information is just valid for that specific use.

[1]PageRank is an algorithm that uses the link structure. Each web is a node and every link is a join, recursively the weight is updated depending on the weights of the pages that links to it. It uses the idea that a link could be understood as a vote from one page to another.

2.3 Related Ideas

In [5] *subgroup discovery technique* is defined. It uses the knowledge of the structure of their topic to do a more accurate Focused Crawler. This is done setting some predicates to represent the relevance clues of the non-visited pages, after that, the classification rules are induced. With this, they design an algorithm that tries to find groups with high number of relevant pages. It is also described the interesting concept of *Relational knowledge representation*: 'linksTo(Y , X, A)' denotes that web page Y links to page X through an anchor[2] element A. With this is possible to establish relationships between pages with any kind of information, this makes possible to model most of the known algorithms just as different relations.

An interesting article is [6]. It says that academic web-pages are better connected to each other than commercial pages. This happens because commercial pages never link to competitors, this is understandable due to they do not want to give any information about other products, that will just happen in the case of bad publicity. In the case of academic, this is not a problem. It says also that the relevance probability is preserved within a radius of about three links, and then it decays rapidly. Finally it describes how being in the vicinity of relevant pages significantly affects the performance of a topical crawler.

A learning focused crawling algorithm could be found in [7]. It explains that it is not necessary to find manually a training set (as most algorithms do). They show a different approach: a TFIDF[3] focused crawling is performed; the information fetched is now used to train a *Naive Bayes*[4] that becomes the real focused crawler. They use the DOM[5] as training set in the second step. This idea seems really useful, then it is used on an algorithm that will be described in the chapter 6.

[2]The anchor text is the visible, clickable text in a hyperlink, i.e. in a web-page it is the underlined-text that can be clicked to lead to another page.

[3]Term FrequencyInverse Document Frequency.

[4]Naive Bayes is a classifier that uses the Bayes's theorem. Therefore it is a simple probabilistic classifier with strong (naive) independence assumptions. It is also called 'independent feature model'.

[5]Document Object Model is a language independent and platform standard object model for representing HTML. It shows in a tree form the structure of a web-page.

2.4 Related Research

A related research is a survey of focused web crawling algorithms [8]. It is a review of the different existing techniques for focused crawling. The techniques are divided into two groups: with and without external help. In the latter it is just an automatic process (without human interaction). The different alternatives are just named and described, any further analysis is done. Thus any conclusion about which is 'best' is reached.

A simple comparison could be found in [9]. It is compared three different algorithms: a neighborhood score that is based on the bayesian method (it gives the same score to all out-links); an anchor algorithm, that uses that piece of text to train Naive Bayes (it gives different score to each link); a mixture algorithm that merges both into a single evaluation function. This last algorithm is related to the one that will be described in subsection 6.6. It also describes that there are subgroups that contain a set of related-topic pages. In this case, the specific topic chosen makes this possible.

A framework to fairly evaluate topic driven crawling algorithms under a number of performance metrics is developed in [10]. It is explained four algorithms: Breath-First Search[6], Shark, Infospiders, PageRank and Best-First Search[7]. The InfoSpiders is an adaptive population of agents search for pages relevant to the topic using evolving query vectors and neural nets to decide which links to follow. This algorithm is explained in chapter 6. The analysis is done over 1000 fetched pages and it is concluded that the Best-First is the best option.

A general evaluation framework can be found in [11]. When a focused crawler is tested several metrics can be used, an estimation of precision, estimation of recall, try to measure the performance/cost form a point of view of bandwidth, CPU use, etc. An automatic analysis is done over 4000 pages per algorithm with: InfoSpider, Best-First Search and Breadth-First algorithms. As conclusion it is suggested to use Breadth-First Search until 700 retrieved-webs because if the seeds are selected properly, any better result will be achieved. In the chapter 8 this will be explained in more detail.

Another paper from the same authors is [12]. A comparison is done between three algorithms: BFS, PageRank and InfoSpiders over 1000 pages. It is concluded that Best-First Search outperforms the other two and PageRank

[6]Also known as BFS: It is a searching algorithm that starts from a node and explores all the neighboring nodes. Later it explores the neighbors of those last nodes. And so on recursively. It works by 'layer'.

[7]Best-First Search is a search algorithm which explores a graph by expanding the most promising node chosen according to some rule, the rule in this cases will be a score.

finished last. An interesting point is how to compare the results, this will be explain in detail in chapter 8.

2.5 Algorithms

In [13], an interesting algorithm is explained, it consists on: Given some seeds (10) it obtains their back-links using a general search engine and this is repeated 4 times recursively. Those 4+1 levels are used to train a learning algorithm (in this case a TF-IDF). Later the Crawler starts with the seeds: download all the children and those children are classified and put in one of the queues (each level, one queue). To select the next document we just pop the one that is in the deeper queue. This is tested with 5000 pages and the results look promising. In chapter 6 will be explained in detail.

In [14], the use of "'tunneling"' is described. This is a technique that tries to avoid the breath-first search and continue in a direction instead of doing a wide search. They conclude that in digital libraries a tunnel with length of 20 is enough, i.e. from a relevant page (they called them *nugget*) it is not interesting to go further than 20 hips because the history path will not give any useful information. It is mentioned that there is a correlation between the score of the parents and grandparents with the children, therefore this information can be used to develop an algorithm. Furthermore it is not just the relation grandparent-parent-children but the path that is followed. The results are quite general therefore it is implemented in this master thesis, in the section 6.5 will be described and explained why it has been selected and how it works.

Chapter 3
Research Question

In this master thesis will try to be found the 'best' driving algorithm for a Focused Crawler. To reach those conclusions it will be necessary:

1. To define a topic:
    - Some related-topic link pages will be found to be used as seeds
    - A term list will be developed to be as a glossary
    - Related and non-related pages will be fetched.
2. To find which is the best way to classify:
    - SVM
    - Neural Networks
    - Bayesian nets.
3. Which are the more worthy algorithms to be implemented.
4. Which are the measures to define what it is 'best'.
    - Precision
    - Time consumption

Chapter 4

Topic Definition

4.1 Introduction

An important issue is how the topics are described and defined. To define a topic is not just to write a title. A title could just lead us to a topic. However, when it gets closer to that concept, when some more descriptions are given, new elements come up, therefore it is necessary to do a more accurate definition.

Another aspect is that setting which idea/concept/topic is not just enough, it is also really important to set the approach, i.e. the point of view from the topic will be defined. An example would be 'microprocessor'. This topic can be seen from different aspects:

As hardware An engineer would see as a circuit, then the definition would be 'a single integrated circuit that has the functions of a CPU[1].'.

As a component A computer enthusiast would see as a part of its machine, the definition of one microprocessor would be 'Intel Mobile Core Duo processor T2300, 1.66 GHz, 2MB L2, 65nm, 667MHz FSB, Socket M 479 pin, Supports Dual-Core'; it seems evident that this approach could be just been good for those that looks that kind of information.

As a subject A student would see a part of a course in its college studies. It would be an abstract concept where software can be executed, the memory controller, the architecture and the instructions would be part of its definition.

Therefore, when a topic is defined it is not just enough with a title, but it needs its context, its aspects, i.e. its approach. Therefore the accuracy is a desirable quality. For these reasons when a topic is defined it is necessary:

[1]Central Processing Unit

- Set a general name for the topic.
- Look for the different approaches.
- Write about the topic in order to clarify the point of view.
- Identify the related topics. This can be used to discard them or to enrich the topic.
- Finding resources to use them as samples.

When all that is done, it is possible to set clearly and to start to define a topic for a Focused Crawler.

4.2 In a Focused Crawler

This process starts as it was described before when a topic is selected and accurately defined. This is the first aspect to be done in a Focus Crawler topic definition.

When that is clear, the second step is to see which are the algorithms will be used. The selected algorithms will be described in chapter 6 and they will set which information is needed. Furthermore, in the analysis (chapter 8) a classification algorithm will be used. Thus, in this case some seeds should be found and how a SVM[2] (it will be explained in section 7.4) and a string-matching (it is explained in section 7.5) are implemented is necessary:

A term list is needed for the string-matching algorithm. This should contain the most frequently terms of the topic and these should be as accurate as possible, i.e. exact match is better than boolean and boolean is better than single words.

Samples to train the SVM: samples of relevant and non-relevant web-pages.

4.3 In this thesis

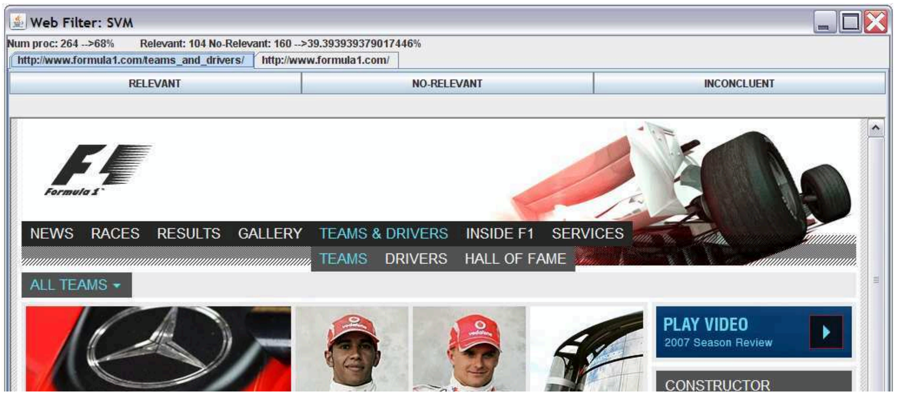
The *Formula 1* topic has been selected to be used in the following parts in the master thesis. The decision is arbitrary since infinite different topics could be chosen. To be as general as possible it is desirable not to be related to digital libraries. It must be a topic with thousands and thousands of entries. It must be a topic the researcher is familiar with (in this case myself); this is a really important issue due to a manual analysis will be done, therefore as much as familiar would help to do a better and more accurate analysis. Furthermore it

[2]Support Vector Machine

must be a non-static topic, i.e. where the pages change eventually. Moreover it must be an interesting topic whose results may be used. However some topics were discarded because they were too popular, an example of this is 'football'; this is because if a brief manual check is done, the sport pages are mostly about football (at least in Europe), therefore the definition topic would be too wide. The results would not be as general as possible and this was a desirable feature.

Therefore, the *Formula 1* was selected because it has all the required features. The second step was to define the approach. After further studying of the topic, the 'Fan of Formula 1' was selected. Actually there is not much approaches to this topic, because the more technical details are industrial secrets that cannot be found in the Internet.

In this light a tool was implemented to find relevant and non-relevant pages. It consists on a window application where web-pages are displayed to the user, then the user evaluates the relevance of the web-page clicking on one of the three available buttons: 'Topic Related', 'Non-Topic Related' and 'Unknown/Error'. A screen shot of this application can be seen in figure 4.1.



```
Num proc: 264 -->68%     Relevant: 104 No-Relevant: 160 -->39.393939379017446%
```

Web Filter: SVM
http://www.formula1.com/teams_and_drivers/
http://www.formula1.com/
RELEVANT
NO-RELEVANT
INCONCLUENT
Formula 1
NEWS RACES RESULTS GALLERY TEAMS & DRIVERS INSIDE F1 SERVICES
TEAMS DRIVERS HALL OF FAME
ALL TEAMS
PLAY VIDEO
2007 Season Review
CONSTRUCTOR

Figure 4.1: Web filter application.

This application is feed with a file that contains web-pages (one page per line), the window described before appears and the selection can start. When a page is evaluated like 'Topic Related', 'Non-Topic Related' or 'Unknown/Error' its URL is saved in different files: 'Related URLs' and 'Non-related URLs'. At the same time the seeds were found. A seed is a web-page that contains a lot of links to topic-related pages and it is a brilliant start for the Crawling. To create the feeding-files a small tool was implemented as well. It uses Google[3] as the source of the results. The first term was obviously 'Formula 1', after that term with some related words in a boolean search. Thousands of pages were selected therefore a filter was needed. To select which pages will be processed, an intersection filter was applied (a web-page has to be in at least two of those Google's results). However to find the non-relevant pages, furthermore of the 'Non-relevant URLs', some pages were included with general queries.

When the URLs to be processed were selected, the manual selection started. The decision of not-using a general collection database is taken to avoid doing just a automatic analysis as it is done in the literature, since if a database is used for training it would not be actually possible to accurately know which were the criteria that make to be or not to be in that collection database. Thus, this is done to maintain the consistence between the training part with the analysis part. It is also clear that an automatic selection with a database would make easier this part and would make possible to select more topics.

As stated before, the topic was really narrowed down, thus for this part of decision it was done with the maximum strictness as possible. The decisions about what it is relevant were:

- Only English web-pages.
- A shop web-page is not relevant.
- Other related sports, such as Nascar, Formula 3, are not relevant.
- A sport page is just relevant if there is mainly Formula 1 information.
- A fan site is relevant.
- A blog is relevant if it talks just about Formula 1.
- Photos and videos are just relevant if they are about Formula 1.
- Technical issues are relevant if they are used in Formula 1 cars.
- A Formula 1 team web-page is relevant.

With all these constrains the filter was executed. They will be used in the same conditions on the manual analysis part (in section 10). This is an important point to keep coherence in the whole research.

[3]www.google.com

To create the term list for the string-matching[4], a focused crawling test database was used. One of the most common meta-data[5] tags is the *key-words*. That has the words that the creator of the web-site thought are more related with the content of its web-page. From the database the most frequently words were saved into a file. A manual selection was done from those words. Finally some words were added from the wikipedia[6] entry: the name of the most famous drivers in the history of Formula 1. Finally manually scores to those terms were given.

In the end of the topic definition several files were created: a file with the most promising seeds, two files with hundreds of relevant and non-relevant pages and two files with terms and their scores (one with the same score for all terms and another manual scored).

[4]http://combine.it.lth.se/documentation/

[5]In the case of HTML, a meta-data is the part of text that is not displayed on the screen directly, it contains information about how text or images should be displayed, it can also contain information about who has created the page, the browser, search information...

[6]www.wikipedia.com

Chapter 5

Architecture

In order to analyze the different algorithms on a real situation we need a Focused Crawler program or environment. Nonetheless none free software or tool was found such as it has already implemented all the algorithms under study, in fact it was not easy to find a free framework. The requirements were to be as powerful as possible (because the crawling process lasts for a long time), but at the same time flexible (to be able to implement the different algorithms), furthermore to be stable, free and it should save all the information of the web (out-links, HTML, tags, URL...). After some investigation, there were two available options: Heritrix [15] and Combine [16].

Heritrix is a web crawler which is open-source, extensible, web-scale and archival-quality. It is implemented in Java. The main interface is accessible using a web browser.

Combine is an open system for web crawling. It can be used for general purposes but also as a focused crawler. It is implemented in Perl. It is controlled from the command-line.

It was decided to use the Combine architecture because it includes functions about focused crawling and because its flexibility takes for granted that any algorithm would be implemented. Furthermore, Perl language has clear interfaces with the databases (an important aspect for the latter analysis) and it is a language that is friendly with the Internet. For these reasons, Combine was chosen.

The Combine's architecture can be seen in figure 5.1. In its implementation just saves the web-pages that fits with the topic definition. Thus, it is not actually a focused crawl process since it does not follow the most promising link. This limitation makes necessary some changes in the architecture (the result of the changes can be seen in the figure 5.2):

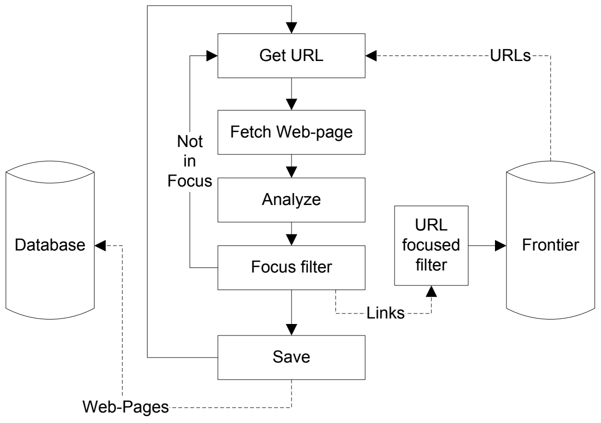


Figure 5.1: Overview of the Combine Focused Crawler.

- The web-pages are always saved, it will be the algorithm that decides which out-links or not will go to the queue and with which score.
- The scheduling is modified in order to have a shorter queue, before it takes all the ready pages.
- Out-links have to be scored.
- When the Crawler starts to run, it has to be differentiated the normal Crawler with the seeds (not scoring) from the score Crawler where the queue is filled with the greatest-scored pages.
- A new initialization was needed.
- The analysis had to be changed.
- Some other small changes were necessary to manage the databases.

The length of the queue is an important issue: On the one hand if it is too short, the crawler has to stop too often to recalculate/fill the queue and it is completely stopped (the frontier has to be sorted). On the other hand, if it is too long it stores many pages with low score that may not be crawled otherwise. In this light the value has to be selected carefully, in both [17] and [10] testbeds were done. In the former it is suggested a length-queue around 200 and in the latter 256. In computer science is often recommended to pick numbers that are power of two, therefore for all testing 256 is selected as length of the queue.

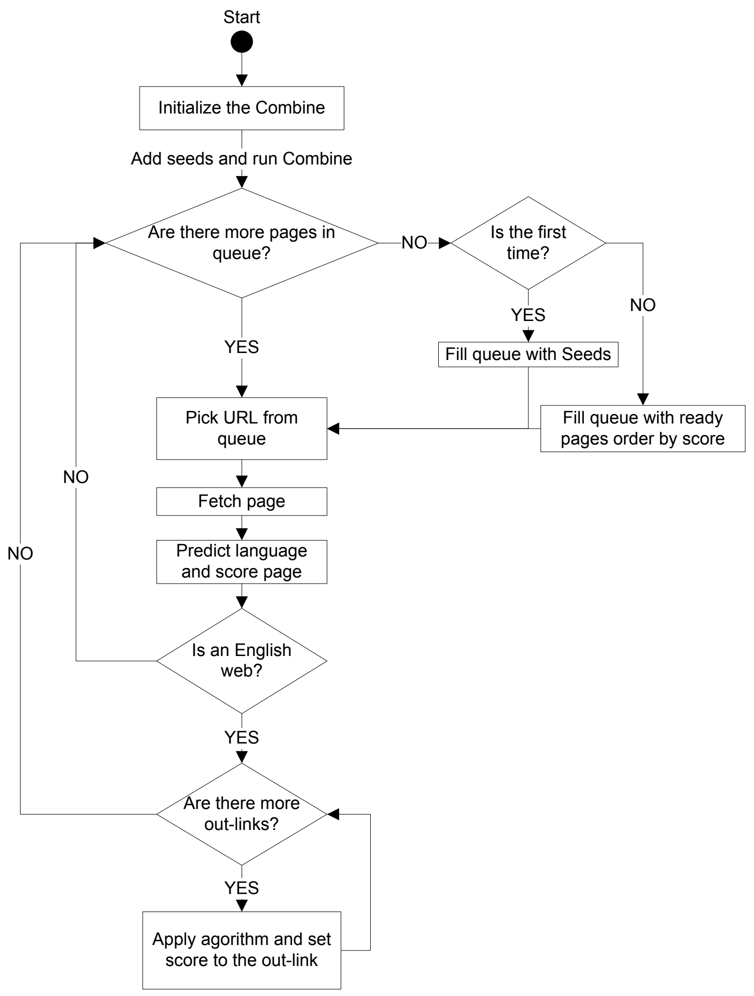

Start
Initialize the Combine
Add seeds and run Combine
Are there more pages in queue?
NO
Is the first time?
YES
NO
Fill queue with Seeds
Fill queue with ready pages order by score
YES
Pick URL from queue
NO
Fetch page
NO
Predict language and score page
Is an English web?
YES
Are there more out-links?
YES
Apply agorithm and set score to the out-link

Figure 5.2: Flow of the modified architecture.

Before the score was only used for filtering (to save or not the information of the web), now it is used as a ranking measure as well. In this point there were two options:

1. The program just saves the necessary information while it is running. When the queue is empty the program rank the pages and insert in the queue the most promising ones.
2. The program rank while it is running. When the queue is empty the program just has to sort the score of the available pages.

Both options work, but since the Crawler is stopped during the refilling-task, it is clear that the second options is the best decision with regard to the processing time. Therefore, in this Thesis, the rank is done while the page is being processed; when the queue is empty it is refilled with the greatest-score ready pages, therefore a sorting algorithm is needed.

Since new data structures were necessary to hold the new situation, it is necessary to change the initialization as well. The score for each link was not possible to be saved in any of the old data tables without deep modification. As it was desirable that the changes become just an add-on, it was decided to create a new table *linkscore*. It contains the unique score for each link that has been found.

In Combine the analysis is the latest step before starting with another web. In that part, it was saved some complementary information like the most probable language of the web (it will be explained how is done in section 7.2). But now it is necessary to have it done in a prior-step. Therefore, it was moved to the algorithms this requirement. Furthermore in Combine the text is trimmed (to 5000 characters), but that makes the algorithm does not work often. A short test was performed about the performance with and without trimming, the results show the same retrieving rate, then the whole text is used for this analysis.

With all these changes implemented on Combine, the design of the algorithms could start.

Chapter 6

Algorithms

6.1 Which algorithms?

The proper selection of the algorithms that will be implemented is one of the main keys of this thesis. A deep research was done to discard as many algorithm as possible (the ones which in other papers are described as worse, see chapter 2). This is necessary since there are too many possible algorithms and inside each algorithm there are a lot of small variants. The selection was performed from the next list (this list comes from [8] mainly):

1. Breadth-First search.
2. Best-First search.
3. Graphic Context algorithm.
4. Path algorithm.
5. InfoSpider.
6. Anchor algorithm.
7. Shark search.
8. Fish search.
9. PageRank.
10. Document taxonomy algorithm.
11. Ontology-based algorithm.
12. Tunneling.
13. Intelligent Crawling.

The 'Intelligent Crawling', 'Document taxonomy algorithm' and 'Ontology-based algorithms' describe just a way how the information can be used for crawling purposes, but in themselves do not describe an implementable algorithm. They were discarded.

The first candidate, the Breadth First, was selected to be implemented due to it is a general baseline. It will be explained in the section 6.2.

The idea of 'Tunneling' that is described in [14] and the idea of 'Path algorithm' described in [9] are quite similar. Thus, both algorithms are implemented as one with the name of 'History Path algorithm'.

In [10] is compared InfoSpider, Best-First, Shark search and PageRank. It is concluded that the best options are: InfoSpider and BF256[1]. Therefore the other algorithms were discarded for the testing. Besides this paper, in [8] the PageRank is said not to be a good Focused Crawler algorithm because it tries to find 'important' pages but not topic-related. The 'fish algorithm' is discarded since it is a simplified version of the shark search.

In [11] is just compared InfoSpider and Best-First. The results show that the best option is Best-First then InfoSpider was discarded.

Finally the ones that have been implemented:

- The Breadth-First.
- The Best-First is a general algorithm that uses an score to define what is 'best'. It was selected because it was the best alternative in most cases in the literature.
- The Graphic Context because it is a promising algorithm that is described in [13]. It outperforms the Best-First in a 12000 fetched-pages test.
- The Anchor algorithm is based on [7] but with different ideas. In the papers it outperforms the Best-First as well. The modifications will be explained in section 6.6.

6.2 Breadth-First

The Breadth-First Search is the simplest algorithm that can be implemented with the Depth-first search. The difference is the kind of queue that is used. In the Breadth-First is used a FIFO[2] instead of a LIFO[3] that is used in

[1]Best-First with a queue length of 256 web-pages.

[2]FIFO: First-In First-Out. The first that is inserted will be the first one that will go out. The behavior is of a line in a shop.

[3]LIFO: Last-In First-Out. The last that has been inserted will be the first will go out. The behavior is similar to a pile of sheets.

Depth-First. In this case, it is much more interesting the FIFO Breadth-First even when the complexity of implementation is the same, this is because if the Crawling starts from really relevant pages and it is more likely that other relevant pages will be close [14]. The BFS is the one that fetches first the closest pages following the order of the minimum number of hips to the root. The pseudocode can be seen in algorithm 6.2.

Algorithm 1 $Breadth-First$

```
$insert\_in\_queue(seeds)$
while $more\_links\_in\_queue$ do
    $link := dequeue\_first\_inserted$
    $doc := fetch(link)$
    $out\_links := extract\_links(doc)$
    $insert\_in\_queue\_last\_pos(out\_links)$
end while
```

6.3 Best-First

Best-First has been studied in [11], [10] and [3]. A rule about what it is 'best' is defined (in most cases a score or rank). When a link has to be selected from the frontier to be fetched that rule is applied, i.e. the 'best' (greater scored/ranked) link is selected. In most cases a classifier algorithm is applied such as Naive Bayes, Cosine Similarity... In this case, a SVM and a string-matching will be used for scoring, the chapter 7 contains the details. The pseudocode can be seen in algorithm 6.3.

6.4 Graphic Context Algorithm

The graphic context is an algorithm that is explained in [13]. In the figure 6.1 is shown the idea of this algorithm. Some relevant pages are found manually and they will be the layer 0 (from this point L0). With the help of a search engine the back-links to those relevant pages are found and they become the L1, again with a search engine are found the back-links of the L1 and they become the L2. This is repeated recursively as many time as we wish (L0←L1←L2...). After the whole process a Context Graph is obtained. It gives an idea of which pages point/lead to the deeper layers in the circle, the closer is to the center the better the page is. This is because if a web-page that is being processed is on L1 means that its out-links (or at least few) will

Algorithm 2 $Best - First$

```
$insert\_in\_ready\_queue(seeds)$
**while** $true$ **do**
  **if** $more\_links\_in\_ready\_queue$ **then**
    $link := dequeue\_best$
    $doc := fetch(link)$
    $score := apply\_rule(doc)$
    $out\_links := extract\_links(doc)$
    $save\_score(out\_links, score)$
  **else**
    $sorted\_links := sort(non\_processed\_queue)$
    $insert\_in\_ready\_queue(sorted\_links)$
  **end if**
**end while**
```

lead to L0 (the related-topic pages), thus those out-links should be fetched as soon as possible because they seem really promising.

The way they implement this idea is using a queue for each layer. A fetched-page is classified to one of the layers and its out-links are added to the queue that belongs to that layer. When a new link is needed, it is taken from the deeper-layer available queue (if it is possible from L0, if not L1, if not L2...) because the closer that it is to the center the more promising its out-links are.

The pseudocode can be seen in algorithm 6.4.

6.5 History Path Algorithm

This algorithm is based in an algorithm that was previously described in section 2.5 and is detailed on [14]. In the paper the distance between relevant with other relevant pages are found. With that information they developed an algorithm that defines 'distance': 0 if the node is a relevant page and 1+parent's distance otherwise. They use that metric to decide which will be the next web-page to be crawled; the conclusion is the better parents have the better children are and that the path history matters. Thus they give a step forward to use this, they change the definition of distance by: 0 if it is a relevant page and

$$distance = min(1, (1-c)e^{2d_p/C}) \tag{6.1}$$

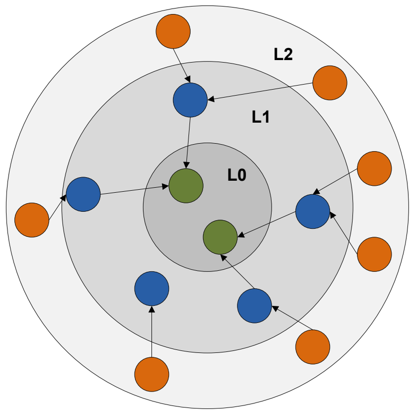


Figure 6.1: GC: The L0 (layer 0) has the manual relevant pages found. With a search engine the back-links are found L1, and it is repeated recursively (L2, L3...)

Algorithm 3 *GraphicContextAlgorithm*

```
$insert\_in\_ready\_queue(seeds)$
while $true$ do
  if $more\_links\_in\_ready\_queue$ then
    $link := dequeue\_best$
    $doc := fetch(link)$
    for $i = 0$ to $num\_layers$ do
      $scoreL[i] := classifyLi(doc)$
    end for
    $layer\_belong := maximum(scoreL[i])$
    $out\_links := extract\_links(doc)$
    $save\_score(out\_links, scoreL, layer\_belong)$
  else
    $sorted\_links := sort(non\_processed\_queue)$
    $insert\_in\_ready\_queue(sorted\_links)$
  end if
end while
```

Where c is the correlation score for the current node, d_p is the parents distance, and C is the cut-off. The cut-off is a number that sets from which distance we do not consider the parent score; in this case they use 20, that was the number that they obtained in previous parts of their research. Thus a nodes distance metric grows exponentially as the parents distance approaches the cut-off and the lower the correlation score of a node, the greater its distance is. This algorithm was implemented in the same way and the pseudocode can be seen in algorithm 6.5.

Algorithm 4 *PathHistoryAlgorithm*

```
$insert\_in\_ready\_queue(seeds)$
while $true$ do
  if $more\_links\_in\_ready\_queue$ then
    $link := dequeue\_best$
    $doc := fetch(link)$
    $score := classify(doc)$
    if $web\_is\_relevant$ then
      $distance = 0$
    else
      $distance = min(1, (1 - c)e^{2d_p/C})$
    end if
    $out\_links := extract\_links(doc)$
    $save\_score(out\_links, distance)$
  else
    $sorted\_links := sort\_decreasing(non\_processed\_queue)$
    $insert\_in\_ready\_queue(sorted\_links)$
  end if
end while
```

6.6 Learning Anchor Algorithm

The implemented Learning Anchor Algorithm uses ideas from two papers: [9] and [7].

In [9] is shown that classification based on the anchor text is more accurate than classification based on the whole page. Moreover, they combine the anchor and the whole parent document (using a Bayesian representation). The conclusion is that the mixture achieves better results. All the results come from an evaluation done about the relation between university pages and researcher pages, i.e. from some Computer Science department pages try to find information about *Courses*, *Department*, *Homepages*, *Project* and *others*. Since most of the university pages have similar web-structure the results cannot be extrapolated to other topics. In the chapter 8 will be checked if just the anchor algorithm outperforms the normal Best-First.

In [7] as explained in section 2.3, the following idea is used: Firstly a normal crawl process is done to collect information that will be used in a posterior training. In this master thesis, a manual and this automatic way will be implemented to check which the best approach is. In this case they do not use just the anchor text, they use the whole DOM structure. Since there is not any explanation why the DOM is better solution than the normal anchor and since in the paper describes before anchor obtains promising results, the DOM structure has been discarded and just the anchor text will be used. This is because there is not reason to think than in a general case (maybe in the university case there is) the web structures are similar to each other. For instance, will a blue letter lead to better results than a red? It will depend on the design of a page, i.e. a web page with red background, red letter will be hidden, thus the letter in that case is in red will not tell any information about if a human may click on that link.

With both ideas a learning anchor algorithm is implemented with several modifications. The pseudocode can be seen in algorithm 6.6.

Algorithm 5 *LearningAnchorAlgorithm*

```
$insert\_in\_ready\_queue(seeds)$
**while** $true$ **do**
  **if** $more\_links\_in\_ready\_queue$ **then**
    $link := dequeue\_best$
    $doc := fetch(link)$
    $svm\_score := classify(doc)$
    $out\_link[] := extract\_links(doc)$
    $out\_anchor[] := extract\_links(doc)$
    $anchor\_score[] := classify(out\_anchor[])$
    $final\_score := svm\_score * factor - anchor\_score * (1-factor)$
    $save\_score(out\_links, final\_score)$
    **for** $i = 0$ to $num\_out\_links$ **do**
      **if** $abs(anchor\_score[i]) > threshold$ **then**
        $add\_to\_training(anchor\_score[i])$
      **end if**
    **end for**
  **else**
    $threshold := retrain\_anchor$
    $sorted\_links := sort\_decreasing(non\_processed\_queue)$
    $insert\_in\_ready\_queue(sorted\_links)$
  **end if**
**end while**
```

Chapter 7

Document topic classifier

7.1 Introduction

Most algorithms described in chapter 6 use an score/rank that measures how relevant a page is. That information will be used as part of the definition of 'best' or it will be 'best' by itself. Therefore this aspect is an important issue that needs deep research. On the one hand if a review about which algorithms are used for this in the literature, most cases use any of the non-learning algorithms because they are well-known and because they are easy to implement. For these reasons in this master thesis will be implemented two completely different approaches:

1. SVM: Support Vector Machine
2. String-Matching

The selection of the latter is based on that it would be interesting to compare a complex algorithm with one of the simplest algorithm of classifying. In this light the String-Matching was selected. Moreover, the Combine framework has this feature already implemented. This comparison (complex and simple algorithms) will permit us to perform a trade-off analysis between the complexity and computational cost.

On the other hand, the selection of the complex algorithm classifier has different options such as Neural Networks, Naive Bayes, IDTF, SVM...

In [18] there is general text classification review where it is compared the ANN[1], the Naive Bayes, Rocchio, C4.5 and SVM. The results show that the

[1]Artificial Neural Network is a computational model based on biological neural networks. It consists of an interconnected group of 'neurons' with inputs and outputs. The network is trained with some known results (inputs and outputs) and it adapts it in order to obtain the same result. This model helps in the cases where there is not a mathematical model for the problem, then if there is just some inputs

SVM outperform any other. From other point of view, in [19] the SVM is compared against a String-Matching algorithm and the SVM outperforms it as well. For these reasons the SVM was selected as classification algorithm. The features of SVM will be described in the sections 7.3 and 7.4.

7.2 Language

The language was an aspect that in the previous testing was not taken into account. After a manual checking about the best pages results, any of them were English, for this reason, a language selection is necessary. This happens because there are some words that are used in different languages (the English language influences other languages very much) and a classification algorithm just uses the words that were found in the training. Therefore when another language page is being processed some English words can increase the score and since there is not any negative word, the score is incredible high. Combine had already included a language predictor, and it is used. Obviously, if all the algorithms use the same rules, the results will show the difference even if the scores may not be accurate.

As said in the chapter 5, some changes were done on Combine framework, basically: the language prediction is done before (its result is now used when the out-links are scored) and the whole document is used instead of 5000 characters to predict. This was decided since the computational cost was almost the same and in some manual verification the results seem clearly better. Thus, in all the comparison the Lingua::Identify[2] will be used.

7.3 Steming

An important question that should be analyzed in any text classification research is if it is necessary to stem or not the words. A *stem* is the part of a word that is common to all its variants (inflected variants). An example would be ’wait’ and its variants: ’waits’, ’waited’, ’waiting’.

In this case we have limited this classification to the English language and since it is a language with low stemming its use is usually discouraged in the literature. In [20] is explained that stemming does not consistently improve classification accuracy and it just happens when a complex stem process is done, in this situation the resources to process a web should be as small as possible and only necessary when a high improvement is achieved. On the

and outputs the problem can be solved (if it has the behavior of the network we have chosen).

[2]http://search.cpan.org/dist/Lingua-Identify/lib/Lingua/Identify.pm

other hand, in [21] is said that in the case of Swedish could be reached an improvement of 4-40% using a general classificator (also in Slovenian and German) but this just happens if the stemming proccess is really accurate.

From other point of view, an analysis about stem on SVM is done in [22]. After testing and analyzing it is concluded:

- In SVM is not necessary to delete the stop-words or rare words.
- It is too much processing time to lemmatizate for not so much improvement.
- It is compared a SVM on text with different kernels[3]: linear, poly and RFB and with different weights (no weights, IDF[4] and redundancy). In a sixteen topics experiment any of those combinations obtains the best score in more than 3 topics, therefore the differences are small (average of 3%).

For these reasons and since this is a comparison, it was decided to use the linear kernel with no weights.

7.4 SVM

Support vector machines are based on the *Structural Risk Minimization*[5] principle from computational learning theory [18]. The idea for classifying text is to see the text as a vector of words, i.e. every different word is a dimension in the vector (it has a position in the vector). As the text has a finite number of words, it is possible to create a finite vector that contains the whole text. As it was said in the section 7.3, in this case a binary SVM will be used. Therefore in the vector will just contain 0 or 1 (0 if there is not the word, 1 otherwise). To see how it works the figure 7.1 represents the simplest scenario with only two dimensions. In the figure the emphasized points are the support vectors (the samples that border the frontier). With those points a hyperplane (in this two dimensions case a line) is constructed to separate the two formed regions.

As in this case is a binary-linear SVM, the prediction operation becomes an addition (instead of a multiplications and additions) due to the coefficients

[3]A kernel is a mathematical function that gives a results from an input. It depends on the kind of the kernel, the result will be linear (linear kernel), poly (polynomial kernel) or RFB

[4]Inverse Document Frequency

[5]Structural risk minimization (SRM) is an inductive principle. It learns from finite training data sets and provides a trade-off between hypothesis space complexity and the quality of fitting the training data.

of the learned vector just has to be multiplied by 0 or 1. Thus the prediction is very fast because it is just the addition of the coefficient terms that the text contains.

The SVM learning can be independent of the dimension of the feature space. This is because its complexity is based on the margin with which they separate the data, not the number of features [18]. Therefore, this implies:

- It is possible to use a really high dimensional input space.
- It is not necessary to select the features, it is possible to use them all.
- Most cases it becomes a linear separable problem.

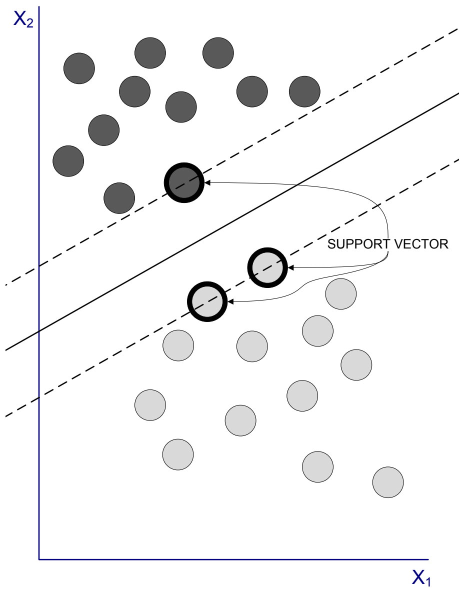


Figure 7.1: SVM: two dimensions and linearly separable.

In our case, the topic definition that was described in chapter 4 is used. The training was based initially on 550 manually selected pages (300 relevant and 250 non-relevant pages). Other 100 pages were manually filtered and the trained SVM was used for verification, it classified 93% of the pages correctly. After starting with the verification part, a final check was done: 40000 pages were fetched using the Best-First algorithm and the best scored pages and pages close to the threshold were manually checked. With regard to the best pages, from the top-200 set only 4 of them (2%) were non-relevant. However the majority of the pages around the threshold were related to *Nascar*[6]. All these new pages in addition to the selected before were finally used to train the SVM with the topic *Formula 1*. This last test is important since some related-topics (but not relevant), in this case an example is Nascar, can have similar words. Thus a final manual filtering is a good solution to make more accurate SVM against similar topics.

7.5 String-Matching

String-Matching is a classification algorithm that is conducted by a controlled vocabulary. The controlled vocabulary has a weight for each word, boolean term or phrase. The algorithm tries to do a string-to-string matching from the vocabulary to the text. When a match is found, the weight of that match is used to increase (positive weight) or decrease (negative weight) the final score. This gives an idea about how close the text is to the class. For this algorithm it is not required any training documents, just the controlled vocabulary.

Different variants of String-Matching are compared in [19]: original set, top 1000 terms from centroid TF-IDF[7] vectors and top 1000 terms from the SVM-normal trained. It is concluded that the best precision is obtained when the original set of terms is used, i.e. only the terms that are presented in the controlled vocabulary. For these reasons, the terms that will be used here are the ones that were defined in chapter 4.

[6]The National Association for Stock Car Auto Racing (NASCAR) is the largest motor sports in the USA.

[7]Term Frequency Inverse Document Frequency. The weight says how important a word is to a document in a class classification.

Chapter 8

Analysis

This is the main point of the master thesis. Up to now, the state of the art has been described, as well as the architecture, the topics have been defined, how to classify the text has been explained and finally the different algorithms have been selected. Now it is time to analyze all the tests that have been done.

In this point all of the mentioned papers describe which process is followed to analysis each algorithm. It usually consists on describing the automatic method selected. In this master thesis this will be done as well, but it would be interesting to perform a manual analysis where those automatic results are compared from a manual perspective, this comparison is one of the motivations and contributions of this work.

Many different measures for evaluating the performance of retrieval systems have been proposed in the literature but there are two especially important:

Precision is the fraction of the documents retrieved that are relevant in the total of retrieved documents.

Recall is the fraction of the documents that are relevant to the query that are successfully retrieved, i.e. the fraction of relevant retrieved pages from the total relevant available pages.

In this case, the total of relevant documents is completely unknown because the Internet is the source, thus it is not possible to define the exact total number of relevant pages on the Web. For this reason the recall will not be used and the precision will be the main performance metric.

As stated before, the definition about which documents are relevant will split the analysis into two chapters: a manual (chapter 10) and automatic (chapter 9) analysis. On the one hand, in the automatic one the cited precision metric will be used from two point of views: with a threshold and with a PDF[1]

[1]Probability density function.

of scores. On the other hand, in the manual analysis will be described how the process has been executed and which has been the results and conclusions that have been reached.

Chapter 9

Automatic Analysis

Automatic analysis means non-manual intervention, i.e. some thresholds are set against the SVM scores and with them the number of relevant fetched pages can be displayed. As said in chapter 8, the precision will be the main performance measure.

In this automatic analysis to decide if a page is relevant or not will be used the text scoring described in chapter 7. A first attempt was to draw a graphic with relevant-fetched pages against the number of fetched-pages, it was quite representative, but one aspect was missing. It was noticed that depending on which threshold was selected, the results change, therefore depending on how high it was set, better or worse results were obtained. Therefore, to solve this issue, it was decided that the best way to show this aspect would be a PDF of the score variable. In this light, with only taking a look at this graphics one understands the distribution of the scores. Drawing an imaginary vertical line in the desirable score, the area that can be seen in the left part would represent the non-relevant pages and in the right part the relevant pages. If that imaginary vertical line is set on the score=0, the results fit with the one that is shown in the graphic described above.

Other option is to use histograms instead of PDF, however, although the histograms are easy to understand they become unreadable if more than one graph must be displayed: it would be necessary to create a 3D graph, this is possible but the results are more complex to read. For this reason, the heights of the bars in the histograms were changed by points and those points joined. If the width of the bars is decreased, the points to be joined are closer, then a more precise line can be drawn. In this case 80 bins were selected, it was a number high enough to keep the shape of the original bars but not too high in order not to have too fast changes. Furthermore the histogram was normalized to area 1 in order to be easier to compare and following the definition of PDF. With this way of drawing, it is possible to overlap different results and they are much easier to compare. On the other hand when the overlapping of PDF is not clear, a *boxplot* will be used. A boxplot or box-and-whisker diagram is a statistical graph where a summarize of distribution of a variable is shown. Their five-number summaries are drawn implicitly: the smallest observation, lower quartile (Q1), median, upper quartile (Q3), and largest observation. A explanation graph can be seen in figure 9.1.

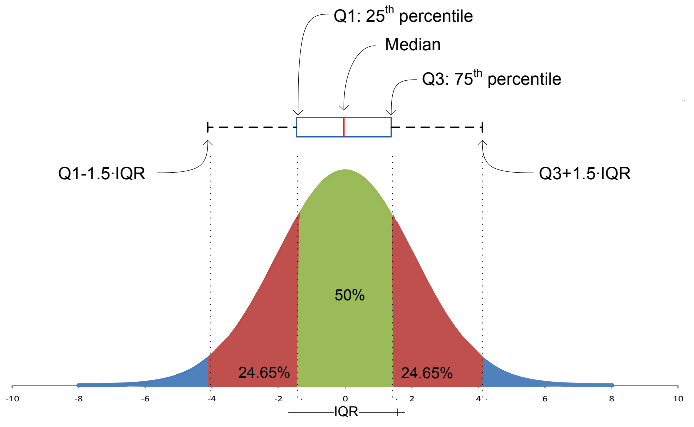


Figure 9.1: Explanation of a boxplot.

Another aspect to be analyzed is if the fetched pages are similar between the algorithms under study or do not since many different variants of the same algorithm were implemented. This question was also addressed from the manual analysis because if the overlapping-pages set is high, we could skip those due to the results would be the same. This comparison is done using the MD5 of the pages, it is consider that two pages have the same MD5, they are the same web-page. This is done because the URL are not unique, therefore two different URLs can point to the same content. Furthermore in Combine the MD5 is used to distinguish if the page that is being crawler is or not already in the database. The overlapping analysis will be done for each algorithm and against each algorithm but just for the more representative ones.

This chapter will be divided into: the part that uses the SVM score (section 9.2) and String-Matching score (section 9.3). After it, a general overlapping will be performed (section 9.5) and some conclusions will be explained (section 9.6).

9.1 Breadth First Search

Before starting with the real test, it is interesting to see how looks like a baseline algorithm, in this case is the Breadth First search that was described in the chapter 6.

Before any modification on Combine, some tests were executed to evaluate the stem and the language issue and their results were used in previous chapters. Nonetheless, when Combine was modified and the BF results were compared to the previous ones they differed, then a further analysis was necessary. The reason for these different results are the way how the Breadth-First is performed. On the one hand the non-modified Combine does not use any algorithm to do a Breadth-First crawl, it is done manually. Some configuration-variables, which Combine has, are used. *AutoRecycleLinks* variable sets if the new links found during the crawl process are added directly to the frontier or are not added. Therefore, if the new links are not added during the crawl process but when all the pages that are in the frontier have been retrieved, it behaves like a Breadth-First search. On the other hand, in the modified Combine a BF algorithm is used. The seeds have score 0, its out-links score 1, the out-links of those pages have score 2 and so forth. When a link is taken from the frontier it is selected the one with the lowest score. At this point the behavior should be the same. The transitions between the layer in the original algorithm are perfect since all the level pages are crawled before starting the next level. However in the new algorithm this does not happen because the webs that have errors and had to wait to be crawled will be finally retrieved later when the Crawler is processing another level. Furthermore when the frontier is sorted, the webs are grouped by the server that they belong, this is done to avoid crawling too many pages from the same server at the same time that may overload it. For these reasons a brief comparison is performed.

The comparison can be seen in the figure 9.2.

The meaning of the titles of each figure are: the first part is 'Big queue' or 'No big queue', i.e. if the network/server lock is checked or is not when the queue is refilled, this avoids overloading the server with too many petitions from the crawler at the same time; the second part is the length of the queue, it defines how many pages are added from the frontier to the queue (the smaller the more often it has to be filled; the longer the lower score of the links is) and the third is if the URLs in the frontier are grouped by server in order to select one per time or not. The one that follows the rules of a nice crawler is the fourth case and it is the most spread one however it is the fastest.

Finally a precision comparison can be seen in figure 9.3. The result is the expected, the one that is more similar to the original Breadth-First Search is better due to its pages are closer to the seeds. And as stated before, relevant pages are more likely to have relevant siblings. Thus the version that its

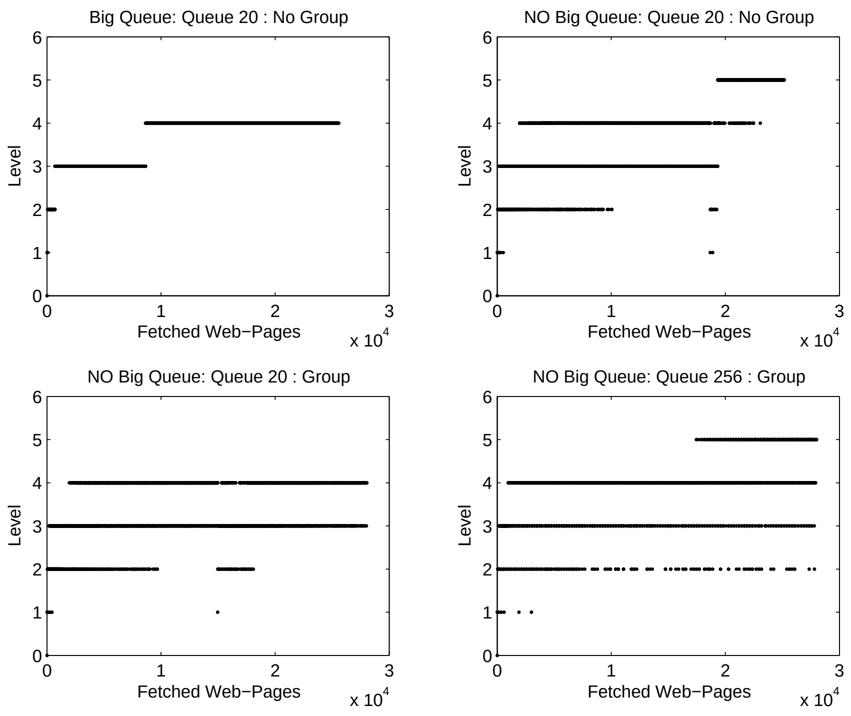


Figure 9.2: Breadth-First comparison: The level/layer for each fetched page.

fetched pages are closer to the seeds should have better results and that is what the figure shows.

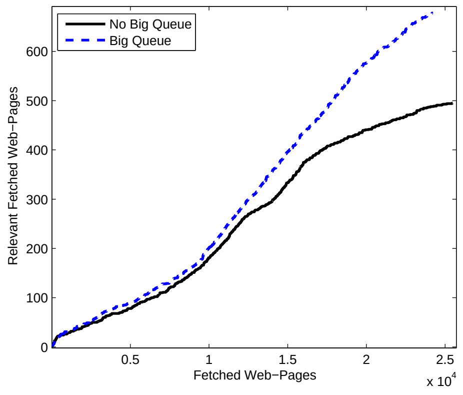


Figure 9.3: Breadth-First comparison: The precision when has big and not big queue.

9.2 SVM

In this section the trained SVM will be used to score the text of the web-pages. Its prediction is a value that generally fluctuates between -1.5 and 1.5 and the greater score is the more relevant the page is.

Initially the Best-First algorithm will be tested. After that, the Graphic Context and History Path will be tested and finally the Learning Anchor Algorithm.

9.2.1 Best-First

This algorithm defines that the 'best' web-pages have to be fetched first. Its implementation consists on giving a score to the links, the links with greatest score are added when the queue has to be filled. In this case 'best' is defined with the SVM score. The more similar the web is to the topic the more score it has, and thus, a better web-page is. Therefore, the next link to be fetched will be the one that has the greatest SVM score in the frontier. Some variants were implemented depending on what was done if a link has already a score. The proposed options were:

- Update the score with the new one.

- Update the score with the mean between the new and the old score.
- Update the score with the mean between the new and the old score but multiply it for a small factor (in order to give advantage to the pages that has double in-links).
- Update the score if the new one is greater.

The best result was obtained with the fourth option, just update if the new score is greater than the old one. With this option a long time test was done and we can see the results of the precision and PDF in figure 9.4.

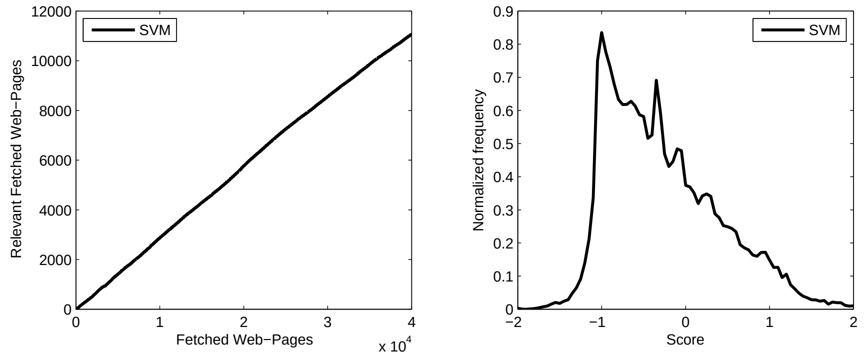


Figure 9.4: Best-First comparison: On the left the precision and on the right its PDF.

9.2.2 Graphic Context

The context graphic has been implemented from two different sources:

- Using the 500 web-pages from the topic definition as the level 0. The search engine Google[1] was used to find the back-links. The level 1 has 400 web-pages, level 2: 300, level 3: 50, level 4: 30 web-pages. It is used for one of the variations the topic definition non-relevant pages as a level 5. In the original paper they say that around 300 webs should be enough.
- Using Entireweb's database[2] is possible to find thousands of back-links for each level, then it is necessary to fix a limit. Since the original paper

[1]www.google.com

[2]http://www.entireweb.com/ and http://www.worldlight.com/

says 300 web-pages per level, the limit was set to 5000 pages. The level 0 has the topic definition 500 web-pages and the other levels have 5000 web-pages. This is because the variant described before fits with the description in the original paper, this wider version may show another perspective of this algorithm.

In the figure 9.5 can be seen the first variant where there are two different tests. The 'original' is described in [13]. The 'Erase' is a test where the links were discarded if they do not exceed a threshold. The results are quite similar but the 'original' shows that is slightly more precise. And both are worse than the Best-First.

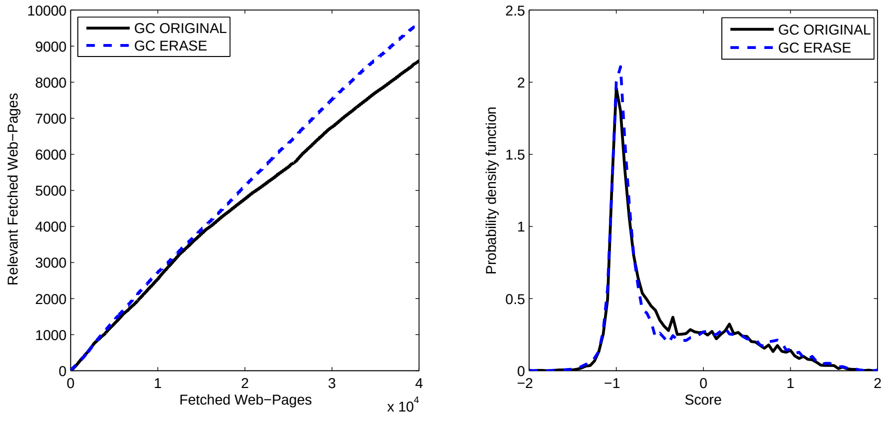


Figure 9.5: GC comparison: The original version of GC and a modification that the links are erased if the score is lower than a threshold.

In the table 9.1 the overlapping between the GC Erase version and the original one is represented. The '1' shows the overlapping between the algorithm and itself, thus it is the 100%. In this case,the overlapping between the original and the erase version is just the 17%.

Table 9.1: Overlapping GC

	GC Original	GC Erase
GC Erase	0.17	1
GC Original	1	

In the figure 9.6 has been represented the precision and its box-plot. There are four versions:

- *'Original'* is the original algorithm described in [13].

- *'W/ non-relevant'* is a modification created by me. The idea comes up since I have hundreds of non-relevant pages that were found in the topic definition. They are pages that lead to non-relevant ones. Therefore they can be used to create a new layer, the one with lowest priority, i.e. the farthest from the center. It should improve the results because it adds information about non-relevant pages.

- *'Erase'* is the variant where the links are discarded if the scores for all the layer do not exceed a threshold. They can be erased because they do not belong to any layer clearly.

The graph shows that the 'w/ non-relevant' is the sightly better than the original implementation. This happens because the new layer gives information about non-relevant pages. On the other hand, the 'erase' version is a bad alternative because it is a 20% worse than the original one.

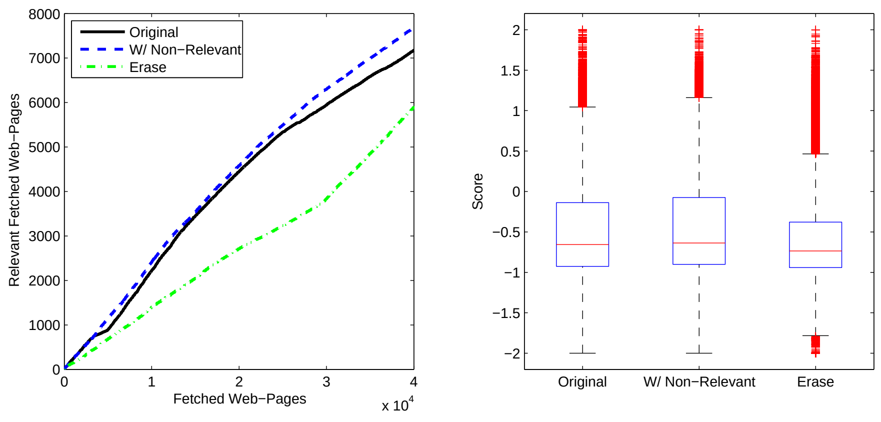


Figure 9.6: GC WorldLight comparison: The 'original' algorithm, a variant where it is used the non-relevant pages to create a layer (w/ non-relevant) and the one where are 'erased' the links if the score is not greater than a threshold.

Table 9.2: Overlapping World

	World Org	World W/Non	World Erase
World Erase	0.18	0.19	1
World W/Non	0.54	1	
World Org	1		

If both alternatives are compared, it is clear that the first implementation outperforms the second one. The reason can be that in the second case to find the in-links was not used the whole URL but was used a sorted version in order to have more hits. This imprecision could lead to the worse results. In any case, it obtains better results including in the modification of the non-relevant pages to create a layer. Furthermore erasing links does not obtain better results.

9.2.3 History Path

The History Path algorithm was described in section 6.5. It was described a formula where the algorithm lays:

$$distance = min(1, (1-c)e^{2d_p/C}) \tag{9.1}$$

And it sets a threshold where the pages that have *SVM score* greater than it, obtain a fixed score. Thus, different variant can be implemented, they depends on the cut-off (C), the threshold and if a link has been or not has been scored previously:

- HP: The original algorithm as is describe in the paper with threshold=0.4.
- HP2: The same as HP but without favoring the pages with more in-links and threshold=0.6.
- HP3: The same as HP but threshold=0.8.
- HP4: With favoring the pages with more in-links and threshold 0.4.
- HP5: The same as HP2 but threshold=0.4.
- HP6: The same as HP5 but instead of scoring with a fixed number when it is greater than the threshold, in this case the score was added to the fixed number.
- HP7: The same as HP4 but threshold=0.8.
- HP8: The same as HP4 but with the cut-off=30 (in the above cases C was 20).
- HP9: The same as HP4 with the cut-off=10.

The selection of the thresholds has been done based on the figure 9.7 since 0.4, 0.8 and 0.6 are the relative minimums in the positive part.

The results can be seen in figure 9.8. If the precision and the boxplot graphs are revised, it is clear that the best is the original algorithm.

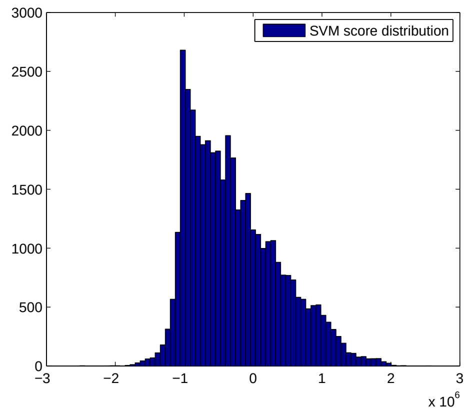


Figure 9.7: Distribution of SVM scores.

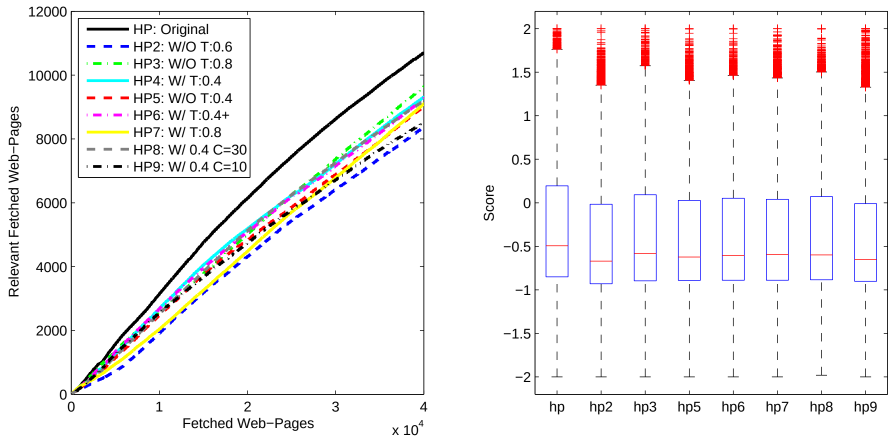


Figure 9.8: HP comparison: The 'original' algorithm and different variants with the threshold, Cut-Off and updating.

Table 9.3: Overlapping HP

	HP Org	HP3	HP4	HP8
HP8	0.14	0.13	0.32	1
HP4	0.14	0.13	1	
HP3	0.29	1		
HP Org	1			

9.2.4 Learning Anchor Algorithm

This is the largest subsection of this chapter because this algorithm is not actually based on any algorithm (we take ideas from two different algorithms), therefore it is necessary more tests in order to find the best options.

The first test is to check if it is a good algorithm, i.e. if it outperforms the Best-First. This is important since the Best-First is one of the baseline. To do this, the learning anchor SVM was trained with the in-link anchors of the 500 pages of the topic definition. Since the anchor most of the times has not information itself (for instance when it is 'press here' or 'this'), it was necessary to use part of the SVM score to complement to the new anchor score. In the figure 9.9 can be seen this. It is represented the different combination of the SVM score and the Anchor score from 0% of anchor and 100% of SVM (the same as the Best-First) to 100% of anchor and 0% of SVM (just the anchor score).

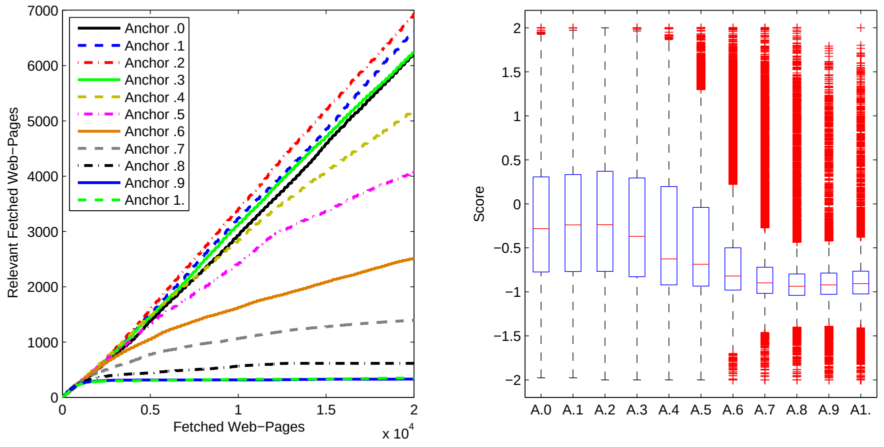


Figure 9.9: Anchor-precision: For instance Anchor .1 is TotalScore = anchorScore 10% + svmScore 90% and so on.

The conclusion is that this algorithm could be a good alternative and also that it is necessary to use high percentage of SVM score to obtain good results, in this case the .1, .2 and .3 outperform the Best-First. Therefore, this will be used in the next parts to avoid useless testing.

The second question to be asked in this algorithm is if the results are improved when it is trained with higher quality relevant anchors, i.e. the more accurate the anchors is the more precise the link is. To test this, the anchor score was predicted from different SVMs that were trained with anchor that link to higher and higher scored pages. The first test was to do a long-term crawl process using as training documents the 500 web-pages of the topic definition. It was used the in-links found in the Graphic Context algorithm to retrieve the anchors that point to those relevant and non-relevant pages. With those anchors a SVM was trained with C=0 [3]. The results can be seen in figure 9.10. The results show that the 0.8 and the 0.9 are the best option (the 0.8 is slightly better if the boxplot is observed).

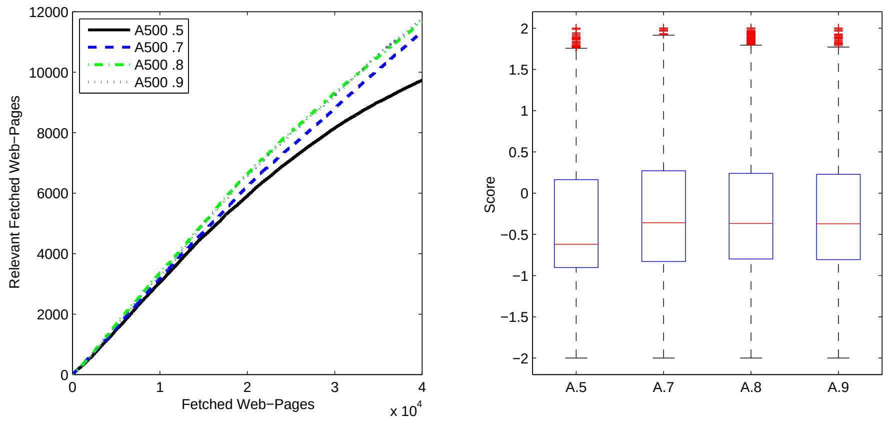


Figure 9.10: Anchor 500: SVM trained with topic definition information.

Table 9.4: Overlapping Anchor 500

	A500.5	A500.8	A500.9
A500.9	0.15	0.50	1
A500.8	0.18	1	
A500.5	1		

[3]C=0 means that the SVM does not penalize any out-place point, it is just discarded. This is done to avoid the over-training.

After a page is scored, the training data is updated if the score is greater than a threshold, i.e. when a page was retrieved and scored, if that score was greater or lower in absolute value than the threshold, its in-links and anchor were taken from the database. Those link-anchors are saved in the training set file. After one run of the previous test was accomplished a file with thousands of representatives anchors was created. In the first test of this new version the threshold was set to 0.4 (the range of the SVM is around -1.5 to +1.5) and the results can be seen in figure 9.11. The results are quite similar to the previous one, that is a good sign because this time the anchor training was done without any information of the topic definition; this means that it can be created from any test that keeps the scores, the links and the anchor text.

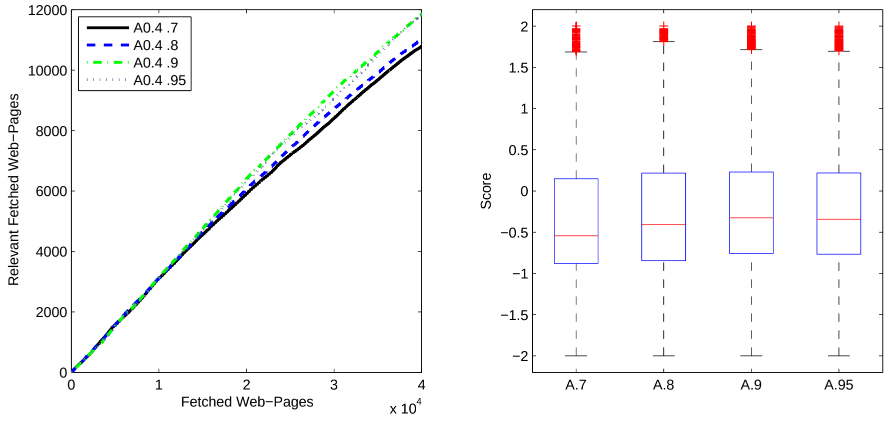


Figure 9.11: Anchor 0.4: SVM trained with anchors retrieved from an previous test with score threshold=0.4.

Table 9.5: Overlapping Anchor 0.4

	A0.4 .7	A0.4 .8	A0.4 .9	A0.4 .95
A0.4 .95	0.31	0.33	0.59	1
A0.4 .9	0.32	0.35	1	
A0.4 .8	0.45	1		
A0.4 .7	1			

The next test was done with a threshold=0.8, the results are in figure 9.12, the results are quite similar but the precision is slightly better in this test.

The former test (with this configuration) uses threshold=1.0. In the figure 9.13 can be seen the results. The precision is better than previous tests and

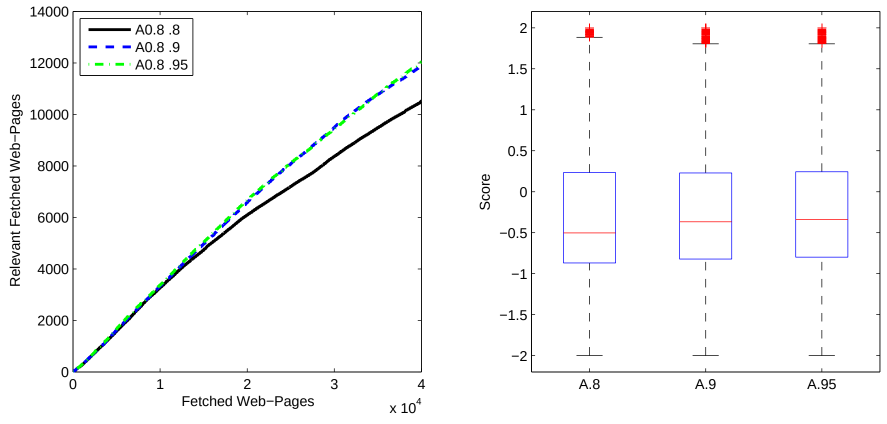


Figure 9.12: Anchor 0.8: SVM trained with anchors retrieved from an previous test with score threshold=0.8.

Table 9.6: Overlapping Anchor 0.8

	A0.8 .8	A0.8 .9	A0.8 .95
A0.8 .95	0.47	0.58	1
A0.8 .9	0.48	1	
A0.8 .8	1		

here the best proportion is .8 and .9 (instead of .9 or .95). In this case it is better if the anchor metric is used more than in the previous case. Thus, more accurate training set has better results and this improvement comes from the anchor score that needs more percentage, i.e. higher scored training leads to better precision.

Table 9.7: Overlapping Anchor 1.0

	A1.0 .7	A1.0 .8	A1.0 .9
A1.0 .9	0.25	0.51	1
A1.0 .8	0.26	1	
A1.0 .7	1		

This last affirmation needs to be checked, therefore the best an the worst training set (threshold 1.0 and the one from the topic definition) was used to train the SVM. If the results are better does not imply anything, but if the results would be worse it can be inferred that better training obtains better results (in contrast to just bigger set for training). As the figure 9.14

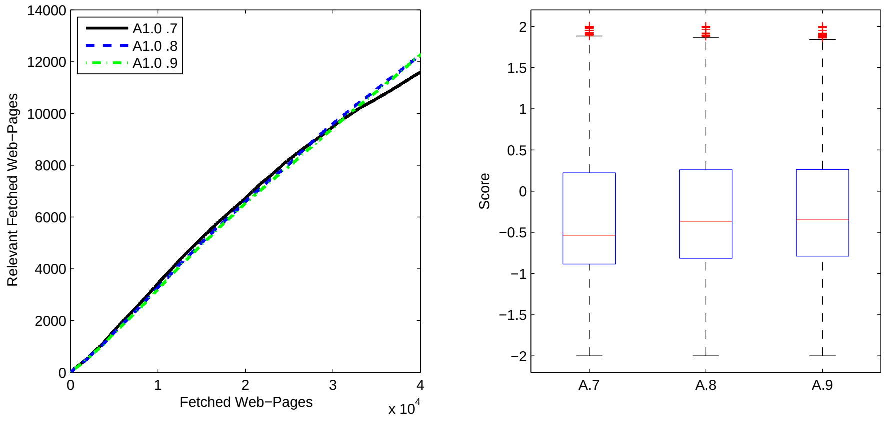


Figure 9.13: Anchor 1.0: SVM trained with anchors retrieved from an previous test with score threshold=1.0.

shows slightly worse results (1%) can be concluded that the last affirmation is correct. However the difference between the alternatives is not significant, this means that the expected improvement in a really long crawl process should be noticeable but in a short crawl does not.

Table 9.8: Overlapping Anchor 1.0 MIX

	A1.0MIX .7	A1.0MIX .8	A1.0MIX .9	A1.0MIX .95
A1.0MIX .95	0.12	0.42	0.55	1
A1.0MIX .9	0.15	0.45	1	
A1.0MIX .8	0.16	1		
A1.0MIX .7	1			

During the testing it was observed that some links have not anchor text, thus it was an interesting question which proportion has anchor text and which has not. To do that, it was used one of the Breadth-First crawl process to check in the databases how many links have and do not have anchor text. The result is that of one million links 796171 have anchor and 203829 do not (around 20%). With this high rate is necessary to see if there is any more information in the links to be used instead. Obviously all of them have, at least, a URL from which the crawler finds the web. Thus a test was designed to check if the URL may help to increase the precision of the anchor text. It was added to the training text the URLs to each anchor line but URLs were cleaned up (such as 'www.', '.com', '.net', etc.). As the best results of the previous test were with threshold=1.0 it was used in this case as well. It is

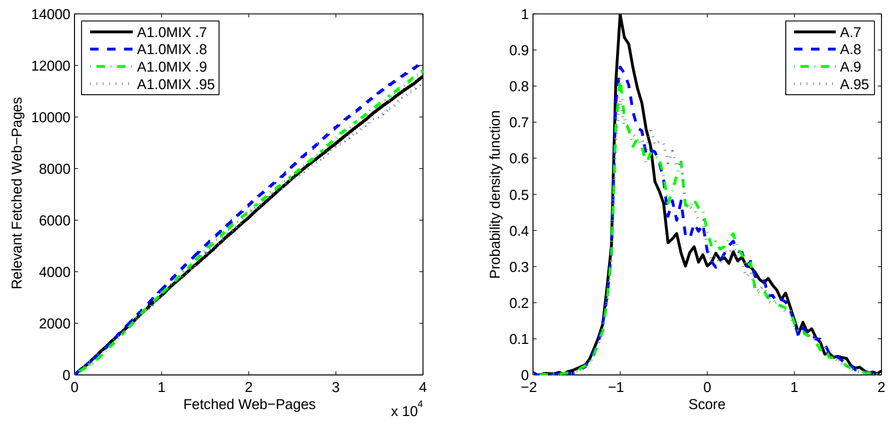


Figure 9.14: Anchor 1.0 MIX: SVM trained with anchors retrieved from an previous test with score threshold=1.0 and furthermore from the topic definition.

represented in figure 9.15. If it is compared with the figure 9.13, this case is worse in a 5.6%.

Table 9.9: Overlapping Anchor 1.0 URL

	A1.0URL .6	A1.0URL .7	A1.0URL .8	A1.0URL .9
A1.0URL .9	0.24	0.29	0.45	1
A1.0URL .8	0.25	0.29	1	
A1.0URL .7	0.39	1		
A1.0URL .6	1			

9.2.5 Automatic SVM Comparison

Before starting with the String-Matching let us compare the best of each algorithm to see if there is a clear winner. In the figure 9.16 this is represented. The baseline algorithm Best-First outperforms all them except the Learning Anchor Algorithm, that it is the one that has highest precision: 10.3% better than Best-First and 14.5% better than History Path.

9.3 String-Matching

In the section 9.2 was used the SVM as the algorithm to score the text, however in this section a String-Matching algorithm will be used to tackle

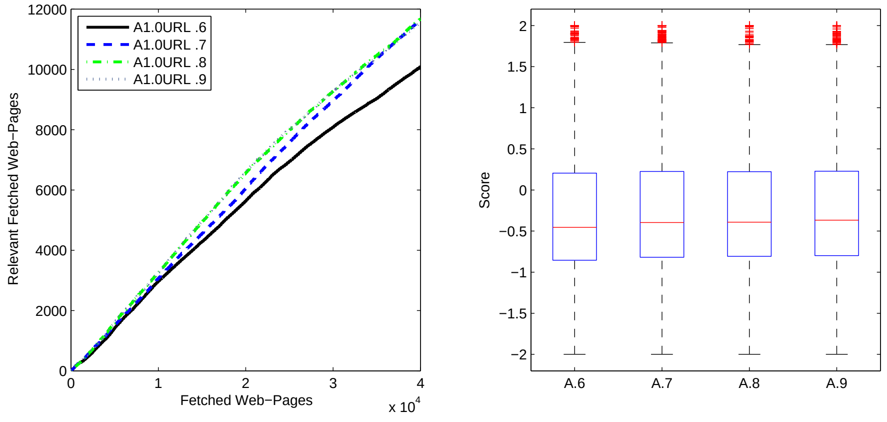


Figure 9.15: Anchor 1.0 URL: SVM trained with the URLs and the anchors retrieved from an previous test with score threshold=1.0.

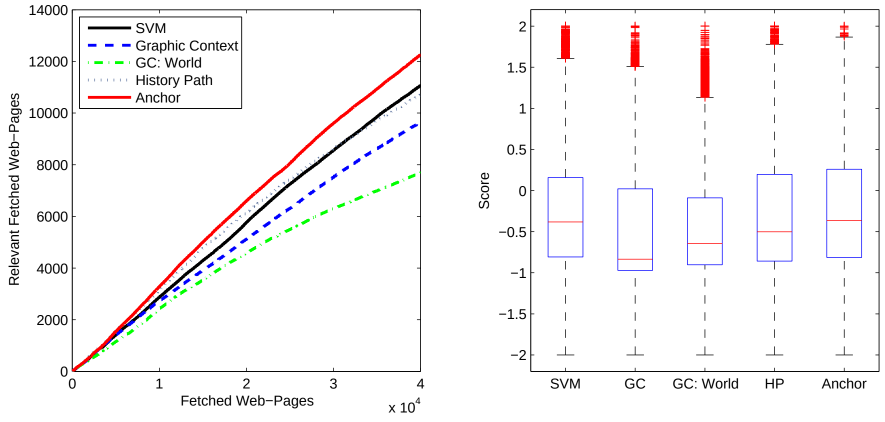


Figure 9.16: SVM comparison: The best of each alternative using the SVM as score.

Table 9.10: Overlapping Best SVM

	SVM	GC	GC World	HP	Anchor
Anchor	0.23	0.10	0.14	0.13	1
HP	0.17	0.12	0.07	1	
GC World	0.12	0.04	1		
GC	0.11	1			
SVM	1				

this issue. The controlled vocabulary that is used as thesauri was described in chapter 4. The focused crawler algorithms that can be implemented are:

- Best-First: It is just to exchange the SVM by a String-Matching algorithm.
- History Path: To exchange the SVM by the String-Matching and normalize the result in order to be able to use the formula described in the paper.
- Learning Anchor: It needs the same changes than the History Path.

The Graphic Context cannot be implemented due to its nature, it would be necessary to create manually a thesaurus for each layer and since the results from the previous section were poor, this algorithm will be skipped.

9.3.1 Best-First

In this algorithm the pages are scored by the String-Matching (from this point it will be called SM). When a page is extracted from the frontier, the one with the highest score is selected. In the section 4.3 was described the process to create the controlled vocabulary but which the scores are is a part that was not described. To do it, it was manually revised term by term. Phrase terms obtain greater scores because they are high quality terms, for instance if the phrase *Fernando Alonso* appears is really representative, however just the term *Fernando* or *Alonso* is not. Thus, two different alternatives are possible:

- Same Score: The same score for all good and bad terms with opposite sign.
- Manually Scored: The scores were set carefully depending on their accuracy and relation to the topic.

The results are shown in figure 9.17. The difference is not significant but if the PDF is observed the carefully scored controlled vocabulary has better results. Therefore, it is not really important how accurate or carefully the vocabulary is scored because the results are quite similar. An interesting aspect is that precision is measured using the SVM score and for being a simple classification algorithm the SM has really good results.

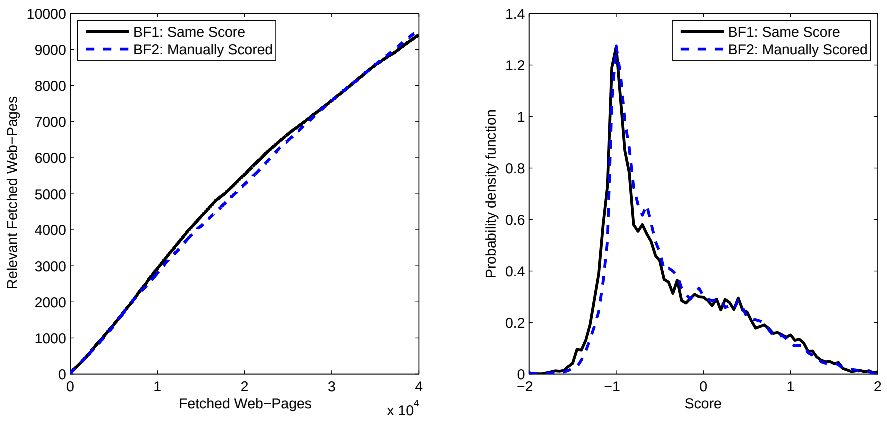


Figure 9.17: Best-First with SM: The general scored controlled vocabulary and one manually scored.

Table 9.11: Overlapping SM

	SM SameScore	SM ManuallyScored
SM Manually Scored	0.40	1
SM Same Score	1	

9.3.2 History Path

The difference with the SVM version is the threshold and the normalization, in the figure 9.18 is represented the distribution of String-Matching scores for a crawling of 40000 web-pages.

To test this algorithm has been selected as threshold 45 and 260 (they are two relative minimum). Furthermore it has been tested increasing and decreasing the cut-off (C). The results can be seen in figure 9.19, the one with threshold=45 is the best option but it is worse than the previous test of SM Best-First. This may happen because when the threshold is low enough it tends to behave as the Best-First, therefore in the best case the results would look like as the SM Best-First.

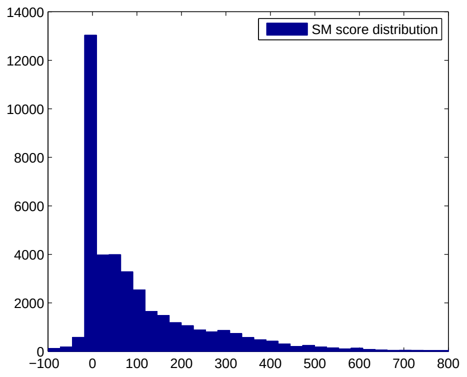


Figure 9.18: Distribution of SM scores.

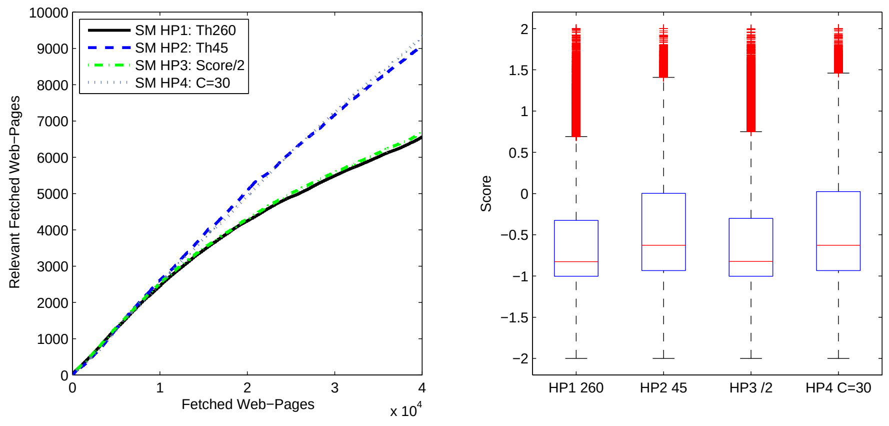


Figure 9.19: History Path with SM: With threshold 45 and 260 and with different cut-off 30 and /2.

Table 9.12: Overlapping SM HP

	SM HP1	SM HP2	SM HP3	SM HP4
SM HP4	0.13	0.59	0.13	1
SM HP3	0.54	0.13	1	
SM HP2	0.13	1		
SM HP1	1			

9.3.3 Learning Anchor Algorithm

It was used the best anchor SVM variation: threshold=1.0, non-URLs and non-mixed. It is checked with different proportions of score since the Anchor score and String-Matching score are not comparable (they are measured with different techniques). Thus, the SM score is normalized and the results with the different percentages are represented in figure 9.20.

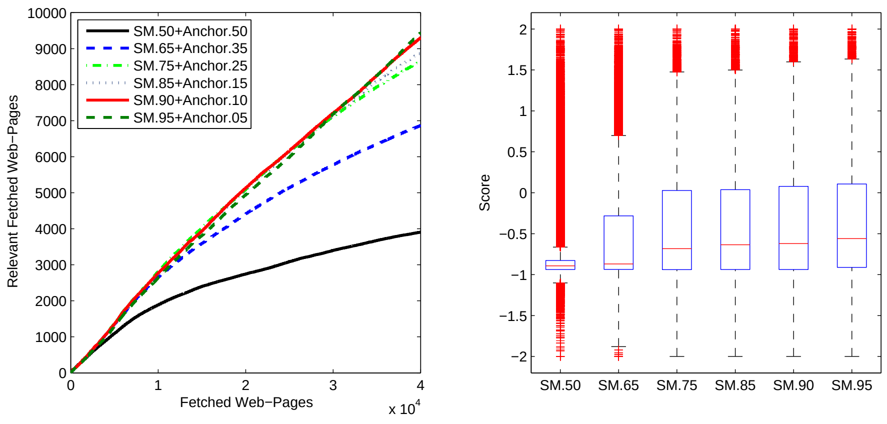


Figure 9.20: Anchor with SM: Different proportion of Anchor score and SM score.

Table 9.13: Overlapping SM Anchor

	SM.75+A.25	SM.85+A.15	SM.90+A.10	SM.95+A.05
SM.95+A.05	0.41	0.44	0.54	1
SM.90+A.10	0.43	0.51	1	
SM.85+A.15	0.50	1		
SM.75+A.25	1			

9.3.4 Automatic String-Matching Comparison

Let us compare the best of each algorithm to see if there is any better than the Best-First algorithm. In the figure 9.21 is represented. In the precision graph the baseline algorithm Best-First outperforms all, however in the boxplot the SM Anchor is the winner, the median is the highest and the distribution tends to be more in the positive scored area. Anyway the difference is not significant. Therefore, if a SM is selected as classification algorithm, it is not interesting to implement a more complex algorithm than the Best-First because the results are not better or not significantly better.

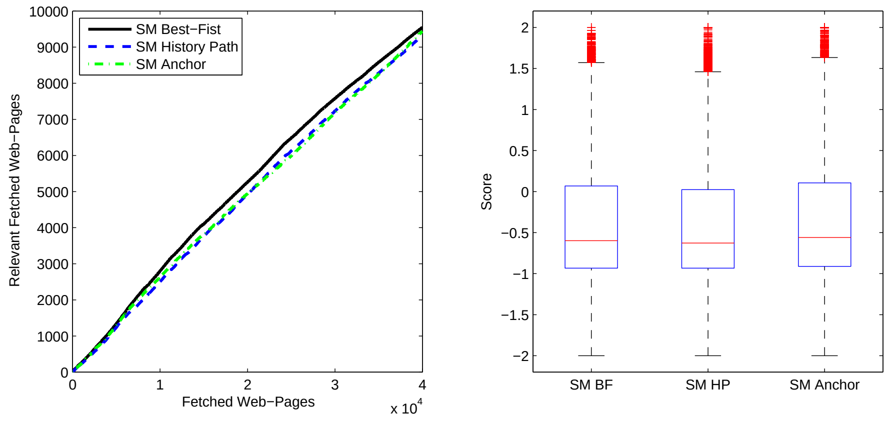


Figure 9.21: SM comparison: The best of each alternative using the SM as score.

Table 9.14: Overlapping Best SM

	SM	SM HP	SM Anchor
SM Anchor	0.30	0.36	1
SM HP	0.36	1	
SM	1		

9.4 String-Matching with SVM

After using the SVM and the String-Matching to score the text and checking that String-Matching is not a bad solution, a good question to be raised would be: how would be the results if both scores were used at the same time in an unique crawl process? This is a brief section where both scores will be joined to create a new algorithm. In this case it is possible to focus into two options: Best-First and the Learning Anchor Algorithm. Since the anchor uses a better text classifier it will be implemented a Best-First with both in a balance importance and in a increasing proportion. On the other hand, in the Anchor variant it will be used the SVM to score the text and the String-Matching to score the anchor. The GC and the HP has shown any advantage and they are skipped. The Anchor option will use the controlled vocabulary but, in this case, some new terms have been included. To add those new terms it was checked the 500 most frequently anchors (in one of the previous test) and they were added manually in the controlled vocabulary, some examples of these new terms are: *advertisement*, *ads* and *football*.

9.4.1 SVM and SM

Firstly it is checked if the new controlled vocabulary obtains or not better results. With the same proportion of SVM score (60%) and anchor score (40%) is tested with the two controlled vocabularies, this can be seen in figure 9.22. The results show that the new controlled vocabulary improves 4% the precision and moreover the improvement is slightly visible in the PDF.

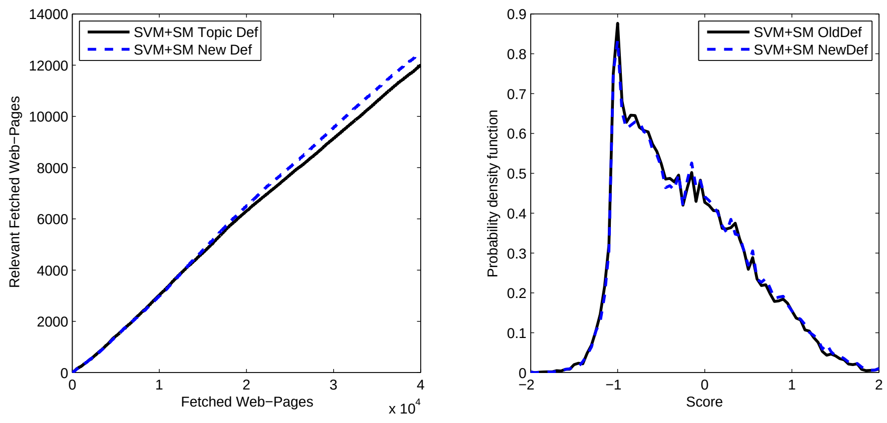


Figure 9.22: Topic Definition comparation: It is compare the controlled vocabulary from the topic definition and one modification (it has been added the most frequently anchors).

Now it is compared with different proportions of each algorithm to find the best result. In the figure 9.23 the tests are shown.

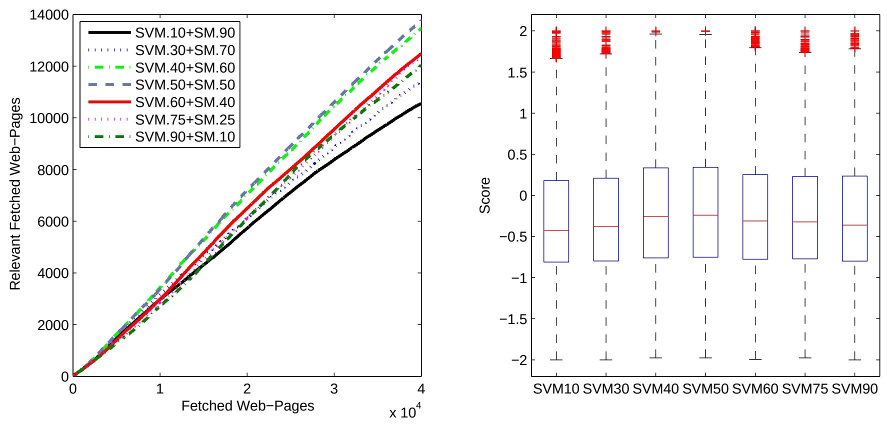


Figure 9.23: SMV+SM comparison: Different proportions of SVM score with SM score.

The results are incredible good. Now it is the best alternative: 12% better than the best SVM alternative (Anchor), 25% better than the Best-First SVM and 45% better than the best SM alternative (Best-First). The reason could be that mixing both scores increases the accuracy of the score, thus, the retrieved pages are those that have a really high SVM score and a really high SM score. As both scores are measured from really different ways, when they are mixed they are complementary. The mixed score is really accurate and its out-links are really promising.

Table 9.15: Overlapping SVM+SM

	SVM.40+SM.60	SVM.50+SM.50	SVM.60+SM.40
SVM.60+SM.40	0.36	0.37	1
SVM.50+SM.50	0.58	1	
SVM.40+SM.60	1		

9.4.2 SVM with Anchor SM

In this case it has been modified the learning SVM anchor algorithm by a SM with the modified controlled vocabulary. This test is performed because the SVM is the best text classifier when a whole text is used but in really sort text as the anchor is, it may not be. Therefore, the general SVM is used to score the text (it will be part of the score) but each out-link will be scored individually using its anchor on the SM algorithm. As the controlled vocabulary does not have all the terms, this algorithm will prioritize the links that are in really relevant pages and whose anchor is in the controlled vocabulary; however links without anchor will be fetched later.

The results are in figure 9.24 and the best alternative is 80% SVM score and 20% the Anchor SM. The results are good: 5.9% more precise than the equivalent with the SVM Anchor but it is worse than the previous one.

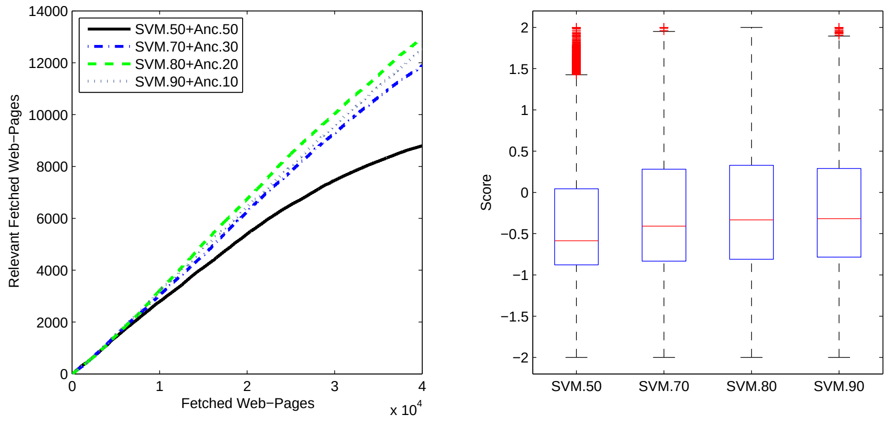


Figure 9.24: SMV with SM Anchor: Different proportions of SVM score with SM Anchor score.

Table 9.16: Overlapping SVM+SM Anchor

	SVM.50	SVM.70	SVM.80	SVM.90
SVM.90+SMa.10	0.32	0.37	0.37	1
SVM.80+SMa.20	0.32	0.52	1	
SVM.70+SMa.30	0.38	1		
SVM.50+SMa.50	1			

9.5 Overlapping

Some of the alternatives have similar results even when they use actually different ways to score the links. Thus, a question comes up: is there a high rate of overlapping between different algorithms? And between the same algorithm but with different parameters? The results of this section can point out how many different algorithms would be necessary to check to do a complete coverage in the manual analysis, i.e. if the overlapping were really high, it would be just necessary to check the most different ones. In previous section was shown the overlapping inside the same algorithm, in the table 9.17 is represented all the overlapping between all pair of the best alternatives for each algorithm.

Table 9.17: Overlapping Best All

	SVM	GC	GC2	HP	Anch.	SM	SM HP	SMa	Svm+Sm
SVM+SMa	0.14	0.05	0.08	0.08	0.14	0.14	0.14	0.15	0.49
SVM+SM	0.13	0.06	0.10	0.09	0.15	0.14	0.15	0.16	1
SM Anc.	0.08	0.03	0.11	0.06	0.11	0.30	0.36	1	
SM HP	0.10	0.03	0.11	0.06	0.12	0.36	1		
SM	0.10	0.04	0.12	0.07	0.14	1			
Anchor	0.23	0.10	0.14	0.13	1				
HP	0.17	0.12	0.07	1					
GC W.	0.12	0.04	1						
GC	0.11	1							
SVM	1								

From the graph can be inferred:

- Overlapping is really low, it is usually around 10%. This means that different algorithms make the crawler behave really different.
- The SM overlapping is slightly higher: around 20%. This means that the SM score leads to more similar pages.
- The highest overlapping occurs between the SVM+SM and the SVM+SMa (they are the best alternatives). This means that the anchor actually has relevant information and it is an aspect to consider. On the other hand the results are better in the SVM+SMa, thus the overlapping might be of the relevant pages.

9.6 Automatic Comparison and Conclusions

As summary of the Automatic Analysis the figure 9.25 shows all the best variants of the algorithms.

The conclusions that can be reached are:

- Most of the algorithms are behind the baseline SVM Best-First search, therefore they are not interesting. The GC and the HP are not actual alternatives in long-term crawl process because the results are poorer than the SVM Best-First and they are more complex (the GC needs five times more computer resources than the other two algorithms).

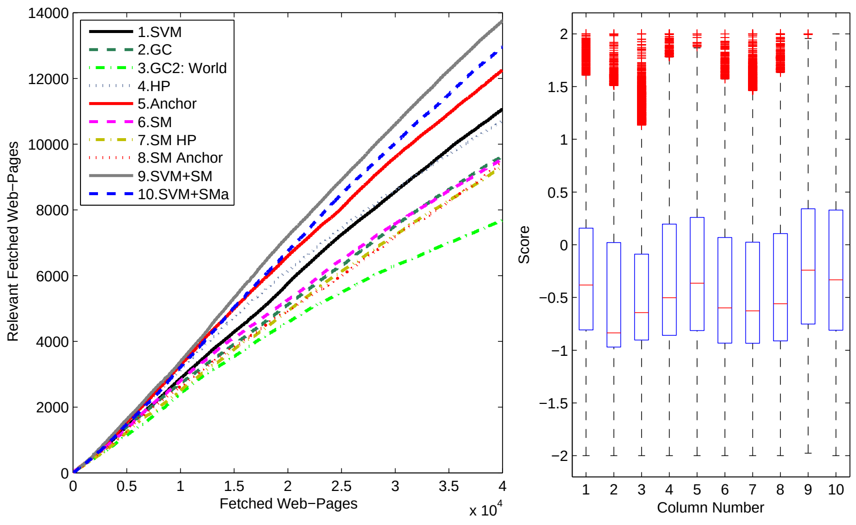


Figure 9.25: Best alternatives comparison: All the best variants of the algorithms: SVM, SM and SVM+SM.

- The Best-First that uses String-Matching could be a good solution if the computational resources are critical. It performs 13% worse than its SVM equivelant but it is just necessary a controlled vocabulary.
- The Anchor alternatives obtain really good results, i.e. to score a link is used the score of the page as the main part but every link is scored individually depending on its anchor. The complexity is higher than the SVM Best-First but the results are significantly better: the SVM anchor is a 10% better than the SVM Best-First and the SM anchor is a 17% better.
- In the subsection 9.2.4 was shown that higher scored anchor training lead to better results, the difference is significant, greater than 5%. It would be necessary a huge crawl process with the learning algorithm described previosly to check if the results improve constantly or there is a limit.
- The best result comes from the mixture of the SVM score and the SM score in balance percentage. It outperforms any other and it is a 24% better than the SVM Best-First. This means that the most important aspect to select the most promising page is the current page (the HP obtained worse results) and if two classification algorithms are joined

the relevancy of the pages are even more accurate and, following their out-links lead to better results.

Chapter 10

Manual Analysis

One of key issues of this thesis is the manual comparison. As it was described in chapter 4 the definition of the topic was done manually to have an iron controlled over the sampling process and to fit as much as possible with the manual analysis. Furthermore this was decided because automatic classification not always leads to precise results because the same classifier used for the algorithms is used for the analysis.

10.1 How-to

In this section the best different algorithms that were proposed will be compared. To fix the sample size, in order to avoid too small and non-representative set of samples, formula 10.1 was followed [23]:

$$n = \left(\frac{\sigma Z_{1-\frac{\alpha}{2}}}{d} \right)^2 \tag{10.1}$$

Where n is the sample size, σ is the standard deviation[1], $Z_{1-\frac{\alpha}{2}}$ is the margin error and d is the size of the population.

In this light, the *size*, the number of web-pages that the crawler could find, cannot be estimated. An important aspect of this number is that it is not critic if is bigger than 20000. A size of 29 billion is cited in [2] was taken. For the other parameters it will be taken: for the response distribution, since any result is known a priori, should be selected the worse case (this is, 50%). The *confidences level*[2] at 5% is a typical value and it was chosen. For the *margin error*[3], another typical value was taken, in this case, 5%.

[1]The standard deviation measures how much the values of a variable are spread.

[2]Confidence level is the amount of uncertainty that can be tolerated, therefore it should be high

[3]Margin error is the amount of error that can be tolerated.

If the numbers are substituted on the formula 10.1 the result is 384'6. Thus, in this entire manual testing part 385 sampling web-pages will be necessary in order to do a proper manual comparison.

About the sampling: Simple random sampling is used, i.e. the same probability to all web-pages is given and with a random variable a correspondence between value-number is set. It is done without replacement.

10.2 Is the automatic analysis representative?

Since manual analysis is much more time consuming than the automatic one and since any other research paper does a manual processing, a first question would be if the automatic method is a valid method of analysis. Here it is not being called into question the effectiveness of the SVM to classify but since two different algorithms are used and since a manual topic definition was done this chapter is actually necessary.

To answer this question and before comparing all the best alternatives, it is checked for the same algorithm, in this case the Learning Anchor Algorithm and all its variants. This will give an idea about if everything remains in place in the manual and the automatic analysis. Using the same algorithm is necessary because the distributions of scores are spread in the same way and this helps to keep the maximum coherence in this part of the analysis.

In the table 10.1 are represented the manual analysis of the different variants of the learning anchor algorithm.

Table 10.1: Anchor manual Analysis

	Algorithm	Expected Result	Manual Result
1	Anchor1.0 .8	34.12%	51.13%
2	Anchor1.0MIX .8	34.67%	47,37%
3	Aanchor0.8 .9	31.26%	46.67%
4	Anchor0.4 .9	32.61%	46.23%
5	A500 .8	31.62%	46.11%
6	Anchor1.0 .7	31.42%	45.36%
7	A1.0URL .7	28.83%	40.13%

In the first column is the rank of the algorithm by the precision, i.e. it is ordered by which has more relevant pages when 40000 pages have been fetched. The second column is the *expected result* and it is the percentage of sample-pages that has score greater than 0. The third column is the *manual result*, i.e. the percentage of relevant pages done by the manual process that has been describe before.

The column *expected results* keeps the order that should be with the exception of the number 1 and 3, this is not significant because it can happen since they are samples. About the *manual results*, that is the relevant column, it clearly follows the exact order than the precision predicted. This is actually important because it can been inferred that the SVM is as good classifier as expected and that a manual comparison is not actually necessary. However, just the order is kept, the proportional distance between variants is not the same, for instance the first algorithm in the precision is just a 1% better than the second, but in the manual result the difference is 3.8%, on the other hand between the 6^{th} and 7^{th} there is 5%, nevertheless, in the precision is just 0.3%. In the other cases, the distance is similar or proportional.

10.3 Algorithms comparison

In the section it will be distinguished between the algorithms that use the SVM and the ones that use the SM because, as it was mentioned before, in the automatic analysis the SVM classifier is used to score the pages for the SVM algorithms but it is, at the same time, used to define the precision. Therefore, the results could show different aspects of this analysis if they are done separately.

10.3.1 SVM

In the table 10.2 are shown the results of the manual analysis of the best alternatives with the SVM classifier. The order is kept and it is similar to the automatic analysis but there are two points where it differs. The GC world algorithm obtains much better results here than in the precision graph, this happens because during the manual selection many Chinese and Japanese Formula 1 webs appeared, since in the other algorithm the non-English pages but clearly related to F1 were accepted (they were a maximum of 5 of 375). In the GC world there were more than the 20%, that algorithm does not outperform any other, this is just testimonial. On the other hand, the Breadth-First does not really belong here, on the SVM algorithms, but it had to be included in any of the manual analysis. The result is almost three times greater than the expected results. As a comment, due to the manual analysis was done in increasing number of retrievement (after the sampling) it showed that the most relevant-pages where found in the first 50% of the selection. Therefore, the Breadth-First is a good alternative for really short crawling, but not for long ones.

Table 10.2: SVM Manual Analysis

	Algorithm	Expected Result	Manual Result
1	Anchor1.0 .8	34.12%	51.13%
2	SVM	31.25%	39.74%
3	GC	25.82%	33.33%
4	HP	25.13%	31.02%
5	GC World	22.47%	27.09%
6	Breadth	4.89%	12.55%

10.3.2 SM

In the table 10.3 can be seen the manual analysis for the best alternatives using the SM as classifier. The order is kept but the percentages are much higher than the SVM cases, the SM obtains much better results that its equivalent in SVM; the other alternatives obtain better percentages.

Table 10.3: SM Manual Analysis

	Algorithm	Expected Result	Manual Result
1	SVM+SM	34.12%	53.87%
2	SVM+SMa	31.25%	50.91%
3	SM	25.82%	47.19%
4	SMa	25.13%	47.02%
5	SM HP	22.47%	41.50%

10.4 Conclusions

Now let us show the whole results of the manual analysis in the table 10.4.

From the table can be inferred:

- The best results in the manual analysis are the same than automatic one: the best alternative is the SVM+SM, the second one is SVM+SMa and the third one is the learning SVM anchor. They are 10% better than the baseline SVM Best-First but this difference in the automatic analysis was wider.
- The SM obtains incredibly high manual analysis results, this can be related to the creation of the controlled vocabulary, those words were the ones that were searched to decide if the page was relevant or not,

thus, since the controlled vocabulary is actually linked to the manual analysis, the results are expected.

- Individually the SVM and SM manual analysis keep the order strictly, thus the method is completely valid when similar algorithm are compared, but when another classifier is used, the results do not match. This happens because to define the precision it is just used the SVM classifier, the improvement that gives the SM is not actually shown; that is just seen in the final results, where both classifiers are joined the results are impressive.

Table 10.4: SM Manual Analysis

	Algorithm	Expected Result	Manual Result
1	SVM+SM	34.12%	53.87%
2	SVM+SMa	31.25%	50.91%
3	Anchor1.0 .8	34.12%	51.13%
4	SVM	31.25%	39.74%
5	GC	25.82%	33.33%
6	SM	25.82%	47.19%
7	HP	25.13%	31.02%
8	SMa	25.13%	47.02%
9	SM HP	22.47%	41.50%
10	GC World	22.47%	27.09%
11	Breadth	4.89%	12.55%

Chapter 11

Contrasting results

11.1 Introduction

After all the development and analysis and before the conclusions it is necessary to check if the results found with a topic like *Formula 1* can be used in a general context. In this chapter another topic will be described and the best alternatives will be compared.

As second topic *microprocessor* has been chosen. It has all the required features that were described in chapter 4: it is not directly related to digital libraries, it is a topic with thousands of entries, I am familiar with it, it is a interesting topic and the pages changes often. Furthermore in that chapter was also described the different perspectives or approaches that this topic has. In this case the hardware view has been selected, thus:

- Shops of microprocessor are not relevant.
- New technologies about microprocessor are relevant.
- News about CPU in computer sites are relevant.
- Features and description of CPU are relevant.
- Courses/subjects about microprocessor are just relevant if they talk about CPU, this means that computer architecture pages are not relevant.
- FPGAs[1] and ARM[2] are relevant if they are used as CPU.

[1] A Field-Programmable Gate Array is a devise used like one-chip programmable breadboard. It has programmable logic components, programmable interconnects and often memory.

[2] The ARM is a 32-bits computer architecture used in a number of embedded designs.

- Embedded systems are relevant if they are described from a hardware perspective.

11.2 SVM

With that definition of the topic, the process to create the training set that was described previously was done. It was found 250 relevant pages and 250 non-relevant pages and they were used to train the SVM.

It was used quickly to compare the algorithms. First it was tested the SVM Best-First. When its results were checked they seem too good, then a further manual validation was performed. When it was done the sampling and the filtering most of the pages were Chinese and Japanese, then the next step was to continue checking through the application that decides which language is. The application does not distinguish those languages, therefore it predicts wrongly that it is English. Since it says that it is written in English, it is scored by the SVM. As it is not actually English, the score is a non-sense that seems random (but usually high). Thus, when a new link is required from the frontier it selects one of the out-links from those random scored Chinese or Japanese that leads to non-English pages that are not topic-relevant.

To solve this issue, it was found in the html-code that most pages uses the charset[3] GB2312 (Chinese), GBK (Chinese) and Shift JIS (Japanese). When the language is being detected it is checked if one of those charset is used in order to avoid being detected as English wrongly.

A new test was run with the new feature and it was checked again. The result was similar because, in this case, most of the pages were in Russian. This happened because the cyrillic alphabet was not detected either. It was again discarded the pages that was the cyrillic charset. In order to avoid any similar problem it was checked the charset-standards and the most popular character encodings such as arabic, greek, baltic were added to be recognized as non-English.

So a new test was done with those constraints and the results were expected to be good this time. A manual checking was done and the proportion of relevant pages was really low and inaccurate. The automatic analysis was revised, it can be seen in the figure 11.1.

It shows that the most pages are relevant (88%) and furthermore the PDF is not like it should be: it should be more in the centre, it should be biased to -1 and it should be more spread. It has to be in this way because the manual checking says that it is really inaccurate and because 88% is too high for the

[3]Charset is a code that pairs a sequence of bits to a character. There are many different charset depending on the alphabet and available character but they have been joined in the Unicode.

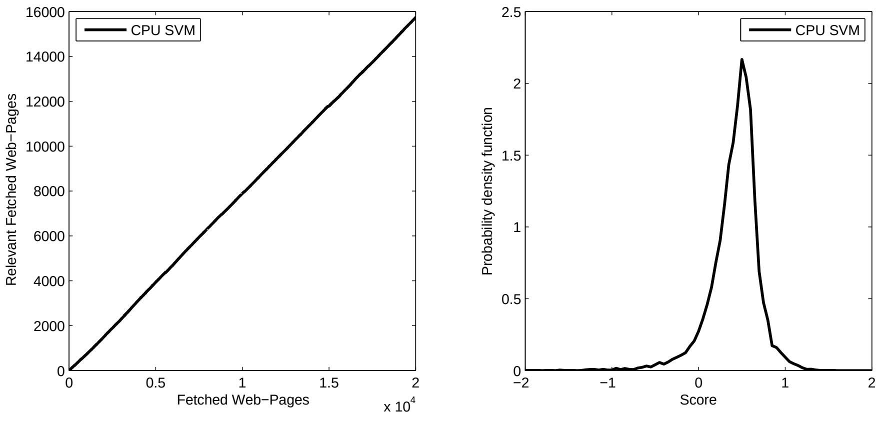


Figure 11.1: First CPU SVM result.

SVM algorithm. Therefore, a intensive training set was required, to do it, it was taken from that SVM crawling the best 200 pages, the worst 200 pages and 200 pages that have a middle scored. About the best 200 pages, there were just 7 that were filtered wrong by the SVM (3.5%) and 10 pages in the worst case (5%). About the middle pages most of them were non-relevant. With this new training set that has 1000 non-relevant pages and over 500 relevant was used with a new SVM.

The parameter C (penalty for unbiased points) was carefully selected, in the table 11.1 can be seen the necessary number of Support Vector for each case. A good way to avoid over-fitting with SVM is to take as less penalty as possible but that keeps the minimum number of support vectors. As it can be inspected, it would be optimal the 0.5 or the 1, thus, it is taken the 0.5 because it is smaller.

Table 11.1: CPU Training, C values

C	SV
0	727
0.1	653
0.5	594
1	593
10	593
100	596

With the new SVM ready, the Best-First test was run again and the results can be seen in figure 11.2.

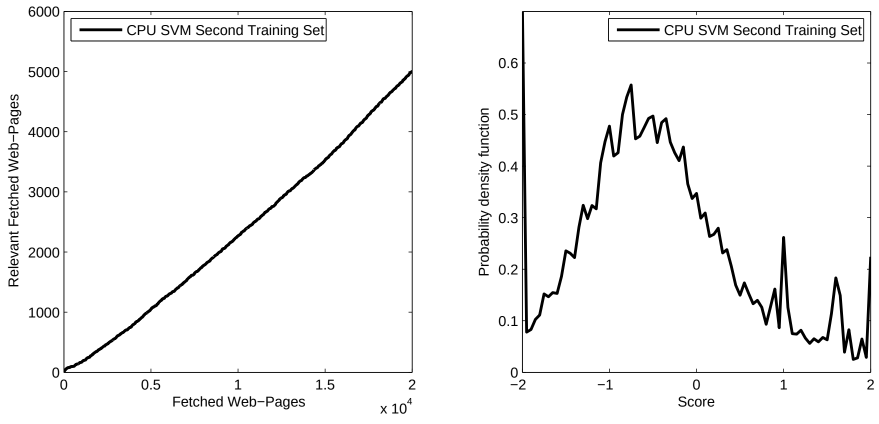


Figure 11.2: Second CPU SVM result.

The results show a proper PDF distribution similar to the *Formula 1* one. Since a wider training set was used and the manual checking shows more accurate results now, it can be inferred that the previous SVM test has not enough samples, that is the reason why it was moved to the right, it was not spread enough and it was not biased to the -1 score.

Now this trained SVM is used to check the best SVM algorithms and alternatives found in the *Formula 1* analysis. The results can be seen in figure 11.3. The figure shows that the best alternative is the SVM and furthermore that the Breadth-First and HP are not alternatives.

11.3 String-Matching

Now the SM and the SVM+SM are tested, it is necessary a controlled vocabulary that is created as was explained in chapter 4. The results are represented in the figure 11.4. The SVM+SM is worse than the SVM. If the *Formula 1* SVM and SM (chapter 9.6) are compare the difference is just 17%, here the difference is 26%. Therfore, the problem is the controlled vocabulary is not accurate enough.

To solve this issue, the weights were tried to be improved as it was done in the *Formula 1* controlled vocabulary. The results were even worse, then it was checked the most frequency terms of the training set and of the fetched pages. It was concluded that terms that were added to the controlled vocabulary were not actually accurate because they were in the relevant and non-relevant

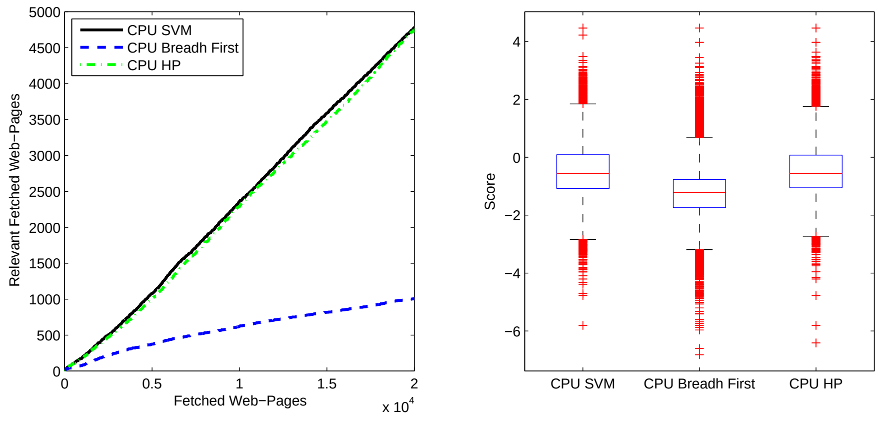


Figure 11.3: CPU SVM results.

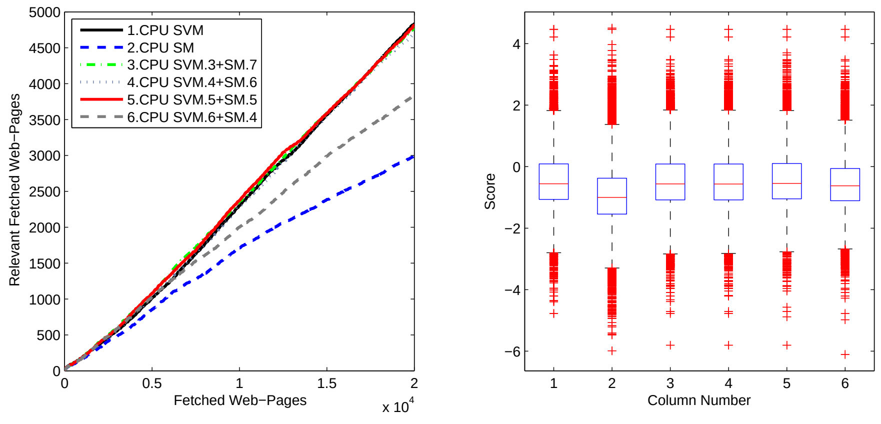


Figure 11.4: Preliminary SM results.

pages. They were removed and it was tested again. The three different controlled vocabularies test can be seen in figure 11.5. The results show that the new controlled vocabulary is a real improvement (around 25%).

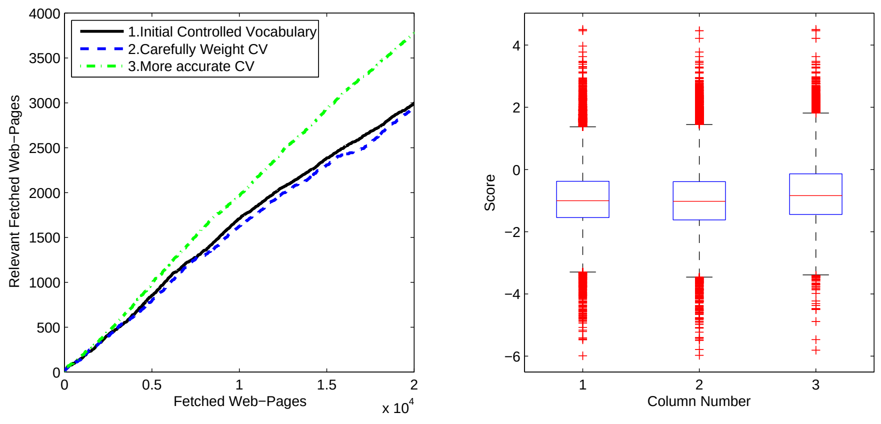


Figure 11.5: Different controlled vocabularies results.

With this new controlled vocabulary, let us compare the SVM+SM to check if the results have been improved as well. It can be seen in the figure 11.6.

The results show a clear improvement and now the SVM+SM outperforms the SVM. In the *Formula 1* was around 26% better, in this case it is just 11%. Therefore it can be concluded that the result is really dependant on the quality of the controlled vocabulary and it can be improved with a more accurate and better weighted controlled vocabulary.

11.4 Summary and Conclusions

There has not been more time to improve the controlled vocabulary or to do other tests. The summary of all best results with the CPU topic is shown in the figure 11.7

.

The conclusions are:

- The SVM can be outperformed by the SVM+SM.
- The Breadth-First obtains the poorest results.
- The results of SM really depends on the controlled vocabulary.

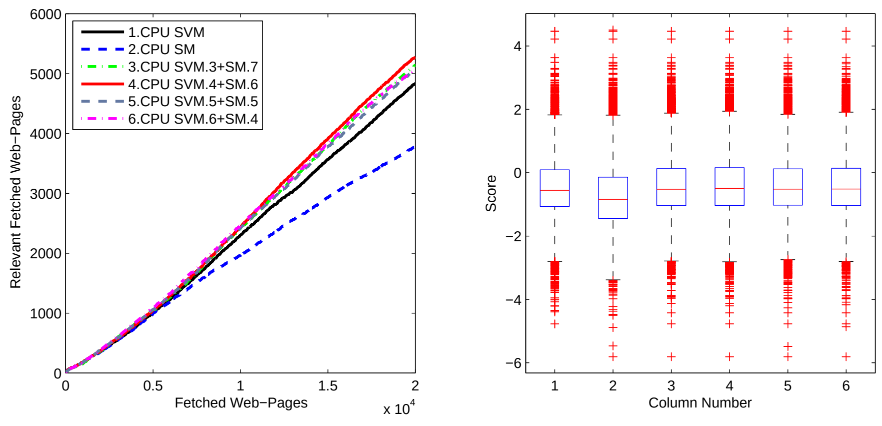


Figure 11.6: SVM+SM results with the new controlled vocabulary.

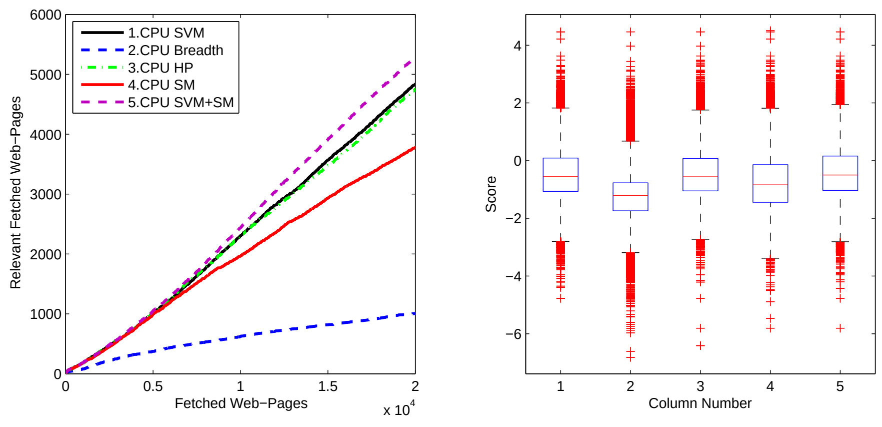


Figure 11.7: Comparison of the best alternatives with CPU topic.

- The SM is a good alternative because it only needs a controlled vocabulary but the results will depend on accuracy of it.
- The HP is a bad alternative.
- The conclusions so far are the same using the topic *microprocessors* than *Formula 1.* Thus, as the topics are not closely related and the results are the same, it can be concluded that the results are general and that it is not necessary to analyze each single topic to find the best algorithm: the SVM is a good alternative and the SVM+SM obtains the best results.

Chapter 12

Final conclusions

- The definition of a topic is a complex task that has to be done carefully by someone who knows what will be expected. If the results are not the desirable, it will help to increase the training set with elements in the limits and in the frontier; that will lead to a clearer PDF and classifier.
- The best results come from the mixture of two classifiers: SVM and SM to do a balanced score for the Best-First search. This happens because both of them contribute to define more accurately what is 'best', they are complementary, thus its children are the most promising ones.
- Alternatives such as Graphic Context and History Path does not show any improvement from the baseline SVM Best-First, this means that the most important aspect to do a focused crawling is to be able to select the best page, because it will lead to the most relevant results.
- The anchor text has almost the same importance than the whole text. In the case of SVM+SM and SVM+SMa the former obtains better precision results in the manual analysis are really close, therefore the anchor text should be considered.
- The learning SVM anchor improves the results when more precise learning set is used. The improvement is not significant enough to outperform the SVM+SM but it lights that the anchor can be used widely to improve the results. Maybe using another SVM is too complex, but the alternative of SM shows accurate results. In any case the algorithms that do not use the anchor could be improved if that information is used. That can be done using a classifier, but the results can be easily improved if the out-links that point to undesirable places are previously discarded (such as 'policy' and 'contact us').
- The language is something that cannot be taken lightly. In this master thesis was used a tool that seemed work properly but when another

alphabet turned up, the results become imprecise and useless. Thus it is necessary to define carefully which the limits of the focused crawling are.

- Unless a really short focused crawling is done where the Breadth-First can be used, the more complex alternatives are justified since the improvement are really noteworthy. The best alternative found is the SVM+SM but if the speed prevails the SM obtains enough good results.
- If the best performance/resources is searched, the SVM+SMa or the SVM Anchor are the best option. As a binomial linear SVM is used to calculate the score, the prediction problem becomes just an addition, however a SM should use a more complex algorithm that can be optimized with KMP[1], but it is clearly slower. Thus, the fastest algorithm would be just two additions, the SVM Anchor, but since the SVM+SMa just uses the anchor with the SM the KMP problem is really small and fast to solve. Thus, any of them are really good alternatives but the SVM+SMa would be optimal.
- The results presented with the topic *Formula 1* are general because when two not even closely related topic were analyzed, both obtained the same results. Thus, it is not necessary to analyze each single topic to find the best algorithm because the same conclusions will be reached.
- The best results turn up when two classifier algorithms are joined, in an ongoing work would be interesting to use and join different classifier to see if the accuracy of the classification and, thus, the improvement of the definition of 'best' appears. On the other hand, it would be interesting to analyze the behavior of a learning String-Matching Anchor algorithm, in this master thesis it has been tested a SM anchor algorithm but there has not been time to test a learning version.

[1]The Knuth-Morris-Pratt string searching algorithm is a String-Matching algorithm that searches for occurrences of a patter, in this case, a word, within a text. It uses the observation that when a mismatch occurs, the word itself embodies sufficient information to determine where the next match could begin, thus bypassing re-examination of previously matched characters

References

[1] ULC, "The google generation: information behaviour of the researcher of the future," *The resercher of the future*, January 2008. http://www.madepublic.com/getdata.php?id=24889.

[2] I. Boutell, "How many websites are there?," February 2007. http://www.boutell.com/newfaq/misc/sizeofweb.html.

[3] L. P. Junghoo Cho, Hector Garcia-Molina, "Efficient crawling through URL ordering," *Stanford University*, 1998.

[4] Z. Zhuang, R. Wagle, and C. L. Giles, "What's there and what's not?: focused crawling for missing documents in digital libraries," in *JCDL '05: Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*, pp. 301–310, 2005.

[5] Q. Xu and W. Zuo, "First-order focused crawling," in *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pp. 1159–1160, 2007.

[6] F. Menczer, "Lexical and semantic clustering by web links," *J. Am. Soc. Inf. Sci. Technol.*, vol. 55, no. 14, pp. 1261–1269, 2004.

[7] S. Chakrabarti, K. Punera, and M. Subramanyam, "Accelerated focused crawling through online relevance feedback," in *WWW, Hawaii*, ACM, May 2002.

[8] B. Novak, "A survey of focused web crawling algorithms," 2004. http://eprints.pascal-network.org/archive/00000738/01/BlazNovak-FocusedCrawling.pdf.

[9] A. Passerini, P. Frasconi, and G. Soda, "Evaluation methods for focused crawling," *Lecture Notes in Computer Science*, vol. 2175, pp. 33–45, 2001.

[10] F. Menczer, G. Pant, and P. Srinivasan, “Topic-driven crawlers: Machine learning issues,” 2002.

[11] P. Srinivasan, G. Pant, and F. Menczer, “A general evaluation framework for topical crawlers,” 2002.

[12] F. Menczer, G. Pant, P. Srinivasan, and M. E. Ruiz, “Evaluating topic-driven web crawlers,” in *Research and Development in Information Retrieval*, pp. 241–249, 2001.

[13] M. Diligenti, F. Coetzee, S. Lawrence, C. L. Giles, and M. Gori, “Focused crawling using context graphs,” in *26th International Conference on Very Large Databases, VLDB 2000*, (Cairo, Egypt), pp. 527–534, 10–14 September 2000.

[14] D. Bergmark, C. Lagoze, and A. Sbityakov, “Focused crawls, tunneling, and digital libraries,” in *ECDL ’02: Proceedings of the 6th European Conference on Research and Advanced Technology for Digital Libraries*, (London, UK), pp. 91–106, Springer-Verlag, 2002.

[15] “Heritrix: Web crawler,” 2007. http://crawler.archive.org/.

[16] A. Ardö, “Combine web crawler,” *Software package for general and focused Web-crawling*, 2005. http://combine.it.lth.se/.

[17] A. Heydon and M. Najork, “Mercator: A scalable, extensible web crawler,” *World Wide Web*, vol. 2, no. 4, pp. 219–229, 1978.

[18] T. Joachims, “Text categorization with support vector machines: learning with many relevant features,” in *Proceedings of ECML-98, 10th European Conference on Machine Learning*, no. 1398, pp. 137–142, 1998.

[19] K. Golub, A. Ardö, D. Mladenić, and M. Grobelnik, “Comparing and combining two approaches to automated subject classification of text,” *Research and Advanced Technology for Digital Libraries*, 2006. 2. http://www.it.lth.se/knowlib/publ/41720467.pdf.

[20] T. Gaustad and G. Bouma, “Accurate stemming of dutch for text classification.”

[21] J. Carlberger, H. Dalianis, M. Hassel, and O. Knutsson, “Improving precision in information retrieval for swedish using stemming,” 2001.

[22] E. Leopold and J. Kindermann, “Text categorization with support vector machines. how to represent texts in input space?,” *Mach. Learn.*, vol. 46, no. 1-3, pp. 423–444, 2002.

[23] R. Lenth, “Some practical guidelines for effective sample size determination,” 2001.