Information Extraction from Web Pages

Tiago Miguel Jacinto Fernandes
tiago.miguel.jacinto.fernandes@tecnico.ulisboa.pt

Instituto Superior Técnico, Taguspark, Portugal

January 2021

Abstract

The content extraction problem has been a subject of study ever since the expansion of the World Wide Web. Its goal is to separate the main content of a webpage, such as the text of a news, from the noisy content, such as advertisements and navigation links.

Most content extraction approaches operate at a block level; that is, the webpage is segmented into blocks and then each of these blocks is determined to be part of the main content or the noisy content of the webpage.

In this thesis, we try to apply content extraction at a deeper level, namely to HTML elements. During the course of the thesis, we investigate the notion of main content more closely, create a dataset of webpages whose elements have been manually labeled as either part of the main content or the noisy content, and apply machine learning to this dataset in order to separate the main content and the noisy content. We proposed an algorithm called X-CEX to solve this content extraction problem, it was based on the Content Extractor Algorithm. Finally, this method and it's processes are evaluated using a different dataset of manually labeled webpages.

Keywords: Information Extraction, Wep Pages, Machine Learning, Supervised Learning, Content Extractor

1. Introduction

1.1. Motivation and Problem

This work is an interesting challenge, as it aims to analyze and improve the existing systems for viewing news, publications, articles, among others, on the Web.

The pages on the World Wide Web, are composed by a lot of irrelevant content called ”noise”, Lan Yi [12] separated this ”noise” into 2 groups according to its granularity, the *Global Noise* and *Local Noise*. Global Noise is related to irrelevant content on pages with high granularity, which, as a rule, is not less than one page. Typically more associated with copied and/or duplicated pages, old unused pages, among others. Local Noise is related to small regions within a Web page, typically associated with advertising, cover photo, navigation links, decorative images, among others. In this work, the focus is to remove local noise from pages with informational content such as news, articles and publications.

Most of the Web pages are composed by irrelevant content that cannot be classified as informational content. This category of blocks is called non-content blocks. Blocks without content are typically advertisements, images, plug-ins, logos, search boxes, navigation links, related links, headers, footers and even copyright information, these blocks are quite common on dynamically generated pages. On the other hand, we call blocks with content (content blocks) to blocks that present the informative content on the page.

The main problem of this work is therefore identified and related to the difficulty of extracting important content for articles and news published on the Web, this is the great challenge behind this project.

1.2. Hypothesis

To solve the problem described above, it is proposed to carry out an analysis of the Content Extractor algorithm, in order to be able to perceive the various necessary improvements to be made, the problems and also the real benefits of this method.

Therefore, in this report, the X-CEX algorithm is presented, implemented by me based on the Content Extractor algorithm for the solution of the mentioned problem. I realized that the Content Extractor is an algorithm in which, despite the good results it obtained and the low computational complexity it has never been explored, there is a very little research and tests applied to this algorithm, which makes it have many points that can be improved.

2. Related Work

The algorithms and methods related to Information Retrieval systems are described in more detail and are more suitable for solving the problem. If it is of interest to the reader, it is recommended that they read more about Information Extraction [4] systems, about the techniques used [11] and still have an overview of the various algorithms and methods that exist [1, 10].

2.1. Body Text Extraction

In 2001 the Body Text Algorithm algorithm was proposed by Finn, Kushmerick and Barry Smith [3]. This algorithm starts by separating the entire content of a page into HTML tags and words, and as a result, the web page is seen as a sequence of bits. The bit is equal to 1 when the respective token is a tag and is equal to zero when it is a word. Therefore, this sequence of bits is represented by the *documentslopecurve* which can be seen in Figure 1, where the x axis represents the number of tokens seen and the y axis represents the number of viewed HTML tags. In this curve we can detect a flat zone highlighted in bold, this zone corresponds to the main body of text, as it is a zone that does not detect any HTML tags.

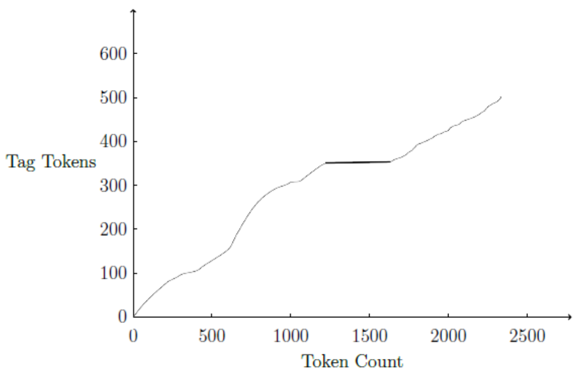

Figure 1: Document Slope Curve

The Body Text Algorithm aims to find a segment on the curve where the slope is smaller, however, this segment cannot be too short, that is, it needs to be long enough to be able to correspond to a long section of text. More specifically, the algorithm intends to find two indexes *i* and *j*, which the number of words found is maximized, while before the index *i* and after the index *j* The number of tags found will also be maximized. This segment can be described by the following equation:

$$T_{i,j} = \sum_{n=0}^{i-1} B_n + \sum_{n=i}^{j} (1 - B_n) + \sum_{n=j+1}^{N-1} B_n, \quad (1)$$

where *N* is the total number of tokens in the document.

The biggest problem with this algorithm is to assume that the main body of text is all together, which is wrong, since there may be several fractions of relevant text separated by the page. However, an improvement has been proposed to this method [8] to search for several main body text segments on the curve, instead of just 1 as originally proposed.

2.2. Site Style Tree

This method proposed by Jadhav and Badhan [5] is based on creating a Site Style Tree from a Document Object Model for each web page, in order to correctly analyze the content of it and to be able to identify the blocks with relevant content.

This method is based on Document Object Model trees and combines them creating a Site Style Tree as shown in Figure 2.

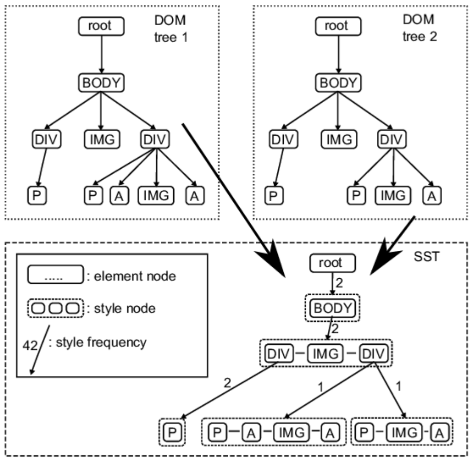

Figure 2: DOM Trees and the final SST

In this figure we can see 2 Document Object Model trees combined, thus resulting in the Site Style Tree where it is possible to see that only the end of the tree differs from the parents, this is because both trees are analyzed and all the trees are combined with common sections, the parts that are not common become another branch of the tree. Two importance values (importance of presentation and importance of content) are used to find the importance of an ”element node”. By this way we find which nodes have the relevant information on the web page.

This method has a great efficiency regarding the presentation style of the web pages and the HTML content. Despite efficiently identifying these blocks with content, this method has very high levels of implementation complexity with regard to the construction of these Site Style Trees.

2.3. InfoDiscoverer

This algorithm, developed by Lin and Ho [6], aims to separate the relevant content from the ambiguous content.

First, the blocks of the pages are extracted based on going through the HTML structure separating all the internal nodes as blocks from the <TABLE> tag. Each block can be a block with content, as these will have one or more strings in their last nodes in the tree, these strings that may be relevant to the pages in question as can be seen in Figure 3. After this first phase, it is necessary to extract the characteristics of each block. In this algorithm, a feature corresponds to a significant keyword. These words are obtained after removing the "Stop Words" and then the Porter Stemming [9] algorithm is applied.

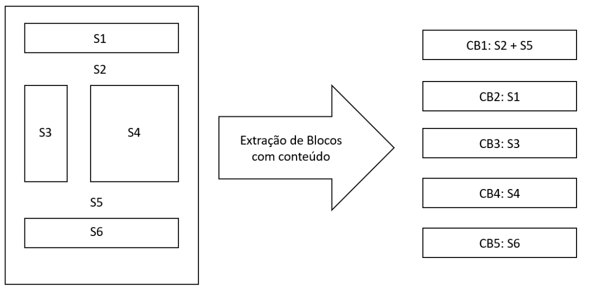

Figure 3: Content Blocks Extraction with strings

After this procedure it is necessary to calculate the entropy values for each of these features according to their distribution, these features will be grouped in a list with a term frequency and its associated weight. Considering this group of features, a matrix of features of the document can be constructed. The entropy value of each of the features can be calculated through the probability of being contained in a line of the matrix.

Then, the entropy value of the block in question is calculated by adding the entropy values of the features of that block as shown in the following equation:

$$H(CBi) = \frac{\sum_{j=1}^{k}(H(Fj))}{k} \tag{2}$$

where F_j corresponds to the feature of a certain block with content (CBi) that contains k features. Therefore, the entropy value of each block is the average of the entropy value of all its features. This entropy value for each block serves to identify them as informative or as redundant over a certain limit. If the entropy value is greater than or close to 1 then this block is considered redundant, this means that these features appear on several pages, if the entropy value is below the limit then the block is considered informative.

This algorithm presents very efficient results when it comes to the discovery of irrelevant content on the page and on the extraction of informative content from the page (for example: news), however, this system has the problem of assuming in advance how to separate the page into blocks, assuming that some of them are the same blocks, but on different pages, so, this algorithm works at the features level.

2.4. Content Extractor Algorithm

This algorithm [2] intends to identify blocks with content from a collection of pages of the same class. First, it starts by separating the entire document or web page into blocks, for that purpose a list of web pages is entered as well as an ordered list of tags separating these pages into a list of blocks and sub-blocks based on the tag set.

With all the blocks identified then you need to start separating these blocks into blocks with no content and content. Those that occur most frequently in the various documents are considered redundant, so they are classified as blocks with no content, that is, the search, advertising, image and other sections, as they end up being very identical on several Web pages, those that appear in isolation are therefore considered as blocks with content, since the text that appears will certainly be exclusive to the page that is processed. The criterion used to distinguish these blocks with content and blocks without content is the Inverse Block Document Frequency (IBDF) which takes the frequency of each block between documents/web pages and then reverse it, this means that, the more frequent a block is, less important content it will have.

To measure this similarity, feature vectors are used, which is, each block will have an associated vector and each vector entry corresponds to a feature, such as, for example, the number of images, the number of terms, the number of scripts, among others. A binary vector is added to the content of the block, on that vector, each entry is associated with a word in the page collection. If the corresponding word exists in the block then the value of that entry will be 1, otherwise it will be 0. [7]

Subsequently the blocks are compared to each other based on their feature vectors, this similarity is calculated based on the cosine similarity and a limit is defined, if the similarity limit is exceeded then the value of block IBDF decreases.

As the blocks go through, their value of IBDF is compared with the predefined limit, if the value is above the limit then the block is considered a block with content, otherwise it will be considered as a block without content and is discarded.

At this moment, the Content-Extractor algorithm is over and the blocks properly separated between relevant blocks and irrelevant to the body of the Web page.

This algorithm produces excellent values of recall, precision and also a great efficiency in runtime, thus detecting the redundant blocks based on the occurrence of these blocks in several web pages, however, if there are web pages with the same style, but with different content this algorithm will not be able to detect them.

3. Implementation

I decided to choose the Content Extractor algorithm as the basis of my work because after a careful analysis of all the algorithms in the Template Detection approach, this was the one that presented the greatest balance in terms of the results of the metrics and level of Implementation complexity. One of the most important factors for this decision was also the fact that it is a little explored algorithm [2] in terms of metrics and heuristics on it.

Firstly, regarding the results presented by the creators [2], it was possible to verify that this algorithm has a very high consistency of Precision, Recall and F-Measure for several datasets, which gave me a lot of guarantees for its effectiveness.

Then, I decided to propose the X-CEX algorithm, the name was created based on its root, since it would be an improvement to the Content Extractor, then the idea of creating an acronym was transforming Extended into X and Content Extractor in CEX.

The X-CEX algorithm will have as input a list of ordered HTML tags and a collection of news web pages of the same class. A class is defined as a Web Site, this means that, this algorithm receives a list of news from the same Web Site, thus guaranteeing that it has the same template or presentation style, so one of the great challenges for this algorithm is to be able to extract the blocks with content for various templates.

The expected output of X-CEX is a list of blocks considered by the algorithm as blocks with content from the web pages of the class provided. It is important to mention that I intend to test the algorithm in classes that are completely different from the classes tested in the proposal of the Content Extractor algorithm, so by this way, I get a more capable method and even with more information.

The X-CEX will be composed by 3 phases, the Block Extraction phase, the block comparison phase and finally the classification phase.

In this architecture 4 we can quickly observe two input arguments, the collection of web pages and the list of HTML tags. In the execution of the X-CEX algorithm we can observe the three phases, the extraction of blocks that in this case play the role of candidates, the comparison between blocks and the classification of blocks as content blocks.

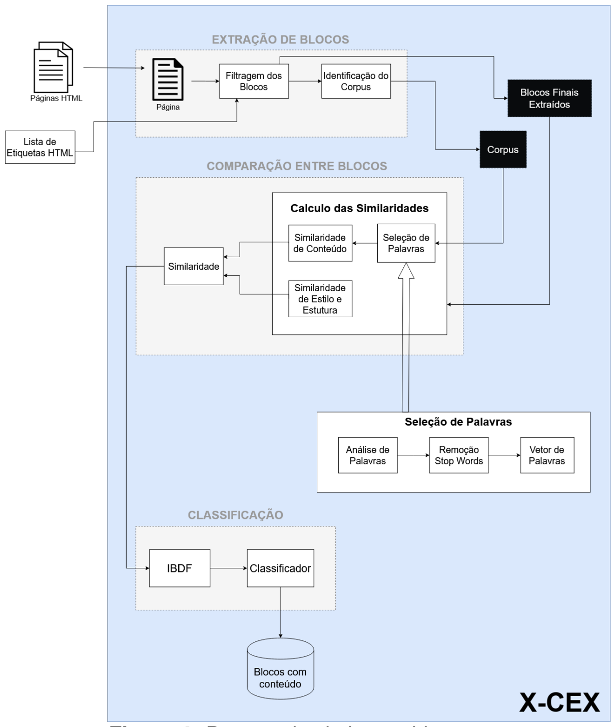

Figure 4: Proposed solution architecture

3.1. Blocks Extraction

The X-CEX Algorithm starts with the block extraction phase (a block is a section of the page that begins with an HTML tag and is closed by that HTML tag). This phase consists of extracting a collection of blocks from all pages provided to the algorithm.

The list of HTML tags is an ordered list created by me based on the visual analysis on several different templates that serves to extract and filter the blocks of the page to be analyzed. The Content Extractor algorithm use the following list of tags: `<table>`, `<tr>`, `<p>`, `<hr>`, `<ul>`, `<div>`, `<span>`, however, and after analyzing, I decided to make the decision to change this list of tags to `<body>`, `<main>`, `<article>`, `<section>`, `<div>`, `<p>`.

3.2. Blocks Comparision

In this phase, the X-CEX algorithm, starts by going through all the blocks extracted from the previous phase one by one comparing them with all the other blocks, in case the block that is compared has an index of very high similarity with another, then the block in question will be prejudiced for the final classification as a block with content or without content.

I decided to propose a system of importance in the X-CEX algorithm, first I separated the comparisons between structure/style of the block and content of the block using a similarity metric for the first component of the block structure and another similarity metric to compare the content (text) of the block. I decided to assign a 30% importance to the similarity of the structure and a 70% importance to the similarity of the text, so I don't completely devalue the structure and presentation style of the block, but on the other hand, I give much more relevance to the content written on it.

The calculation of this similarity is then carried out using the following equation:

$$Sim(Bi) = (SImp * SSim) + (CImp * CSim), \quad (3)$$

where SImp is the importance of the structure, SSim is the similarity of the structure, CImp is the importance of the content and CSim is the similarity of the content, obtaining a result between 0 and 1, with 0 means the most different possible and 1 the most similar possible.

Having the result of this similarity, it is then important to define a treshold at the beginning of the algorithm for the result, if the value is higher than the limit then the two blocks are considered to be similar, otherwise, the two blocks are considered to be different.

To calculate the similarity of structure and style between two blocks I decided to use the already implemented HTML-Similarity, this uses the tag sequence comparison HTML for calculating structure similarity. To calculate style similarity, this method extracts all CSS classes from the blocks and then applies Jaccard similarity to the names of these classes. The result of the similarity of structure and style is a combination of these two similarities, both have an equal importance.

To calculate the similarity between two blocks in terms of their content, the cosine similarity measure is used with the same reasoning as the Content Extractor algorithm, that is, each block is assigned a binary vector, and each entry of this vector corresponds to a word, if that word exists in the block text then that vector entry is 1, otherwise the vector entry will be 0 calculating finally the cosine value between these two binary vectors.

It is important to note that the words taken for the creation of this binary vector are all the words in the collection of extracted blocks, therefore, all vectors of all blocks will have the same dimension so they can be compared.

The cosine similarity is applied to two vectors in a certain dimension, the result of the similarity is nothing more than the cosine of the angle made by the two vectors, however, this angle with big and different vectors is not easy to know so the following formula can be used to determine this similarity:

$$Sim(A, B) = \cos(\Theta) = \frac{A \cdot B}{\|A\| \, \|B\|}, \quad (4)$$

3.3. Classification

At the end of the analysis of each block, IBDF takes into consideration [2]. The IBDF is the inverse of the frequency which a block appears on the various web pages, that is, in practical terms, the more pages that have a block similar to the block that is compared, less relevant this block will be. This means that if a block is repeated/has similar blocks on several web pages then that block will not be a content block.

Each block has an associated IBDF value that starts at 1 at the beginning of the comparison phase and after it has been analyzed on all Web pages and checked how many of them have a similar block present, then, the IBDF value of the block will be recalculated, and the more pages that have at least one block similar to itself, the worse it will be.

This means, the IBDF of a block is nothing less than the inverse of the sum of the number of pages that have at least one block similar to itself as shown in the following equation:

$$IBDF_i = f\left(\frac{1}{|S^i|}\right), \quad (5)$$

where S_i is the sum of the number of pages as shown below,

$$S^i = \cup \left\{P_l : Sim(B_i, B_k) > \varepsilon, \forall B_k \in P_l, \forall P_l \in S\right\}, \quad (6)$$

with S being the collection of pages from the same source, B_i and B_k the compared blocks and P_l is the added page.

In order to be able to classify a block as a content block, it is necessary to define a treshold for this block IBDF value. I have defined for this algorithm that if a block has similar blocks on 20% of the pages then this block is no longer a relevant block to the context of the problem and is classified as a non content block. For example, in a collection with 100 Web pages, if a block has similar blocks in less than 20 pages then that block will be a content block generated by the algorithm.

As the value of IBDF is not exactly the number of pages, but the inverse of the number of pages with similar blocks, then the limit of IBDF will also be this inverse, but applied to precisely 20% of the total number of pages as indicated by the following expression:

$$Treshold - IBDF = \frac{1}{|0,2 * N|}, \quad (7)$$

where N is the total number of pages in the collection.

4. Results & discussion

4.1. Datasets

The Content Extractor algorithm only extract news pages from English Web Sites, however, in this work I intend to evaluate the X-CEX algorithm with other languages such as Portuguese, Spanish, Italian and French, in order to test its effectiveness with different languages.

7 Web Sites were chosen to be analyzed with the X-CEX algorithm, these are the Portuguese newspapers A Bola, Record and Diário de Notícias, the Spanish Marca and AS, L'Équipe, which is the only French and to finish the only one Italian newspapper is Corriere dello Sport. 160 HTML pages were extracted from each of these Web Sites, so, in total, I have 1120 pages divided into 7 different templates. These pages were divided into 2 dataset types, the training set and the test set. The first one serves to implement the improvements and trying until obtain the best results possible, the second one serves to test the algorithm. I divided with 75% of the pages to training set and 25% of the pages to test set.

4.2. Evaluation Metrics

The metrics used to evaluate and compare this work were Precision, Recall, F-measure and Accuracy. First, it should be explained that in this context the values of precision and recall are based on the 'items' considered as true positive, false positive, true negative and false negative, in practice, these groups intend to divide and display the 'items' that have been evaluated correctly and the 'items' that have not been evaluated correctly. The group of positives represents the blocks extracted by our work as relevant, and may have been well or badly extracted (True or False), the same reasoning is applied to the negatives, these being the blocks that were considered irrelevant by the work developed, also these divided in good or bad (True or False).

Additionally, I tested the algorithm in execution time too, I wanted to analyze the time that this algorith takes to execute in each template.

4.3. Tests & Results

We can see that, the algorithm works very well for AS and Diário de Notícias templates obtaining values of Precision, Recall, F-Measure and Accuracy above 90% which shows to be very reliable in these same Web Sites. The Marca and Record end up representing an excellent Recall, which means, obtaining almost all content blocks of the news, however, they end up including many irrelevant blocks decreasing the Precision value. L'Équipe and Corriere dello Sport, on the other hand, show themselves in the opposite direction obtaining an excellent Precision and low Recall, meaning that practically all the blocks extracted are relevant, however, some others remain to be identified. At both sites, these results can be justified by their block extraction phase.

It is also important to mention the excellent execution time of the algorithm, where the Diário de Notícias website only stands out in this field. This increase in time can be justified by the fact that the blocks of text in this template mostly contain many sub-blocks (such as, for example, bold, quotes, italics, etc.), which makes them more complex for comparisons performed on the algorithm.

We can also observe that, Accuracy shows to be very in tune with Precision, this is justified by the fact that Accuracy is based on the number of blocks extracted correctly (both with content and without content), whereas Precision, is only based on the blocks with content extracted correctly, which shows that content blocks and non content blocks are generated by the algorithm in a very proportional way compared to the expected.

Overall, I think I managed to get an excellent algorithm here capable of extracting the relevant content for several different templates, excluding the A Bola website, the worst F-Measure result is 80 % for the Corriere dello Sport website, which shows that there is a great balance in this algorithm between Precision and Recall. The languages used in these templates does not seem to have a great importance to the results, so it seems to be a component already solved previously by the Content Extractor algorithm.

Comparing X-CEX with the base algorithm, Content Extractor, we can see that there is a clear improvement for AS and Record as shown in Figures 5 and 6.

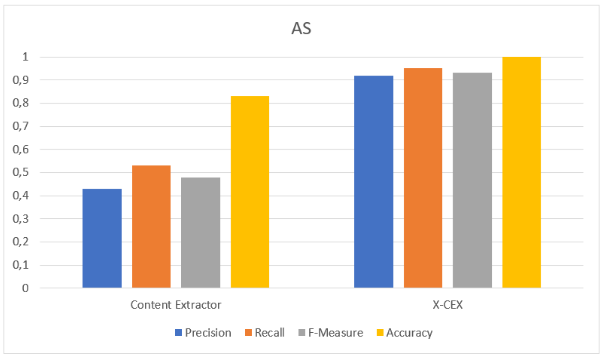

Figure 5: Comparison between X-CEX and Content Extractor for AS

In this case, we can see that the four metrics shown in the images are closer so more cohesive are the results, so in the case of AS and Record we can see a great improvement of the results. On the other hand, we observed that for Corriere dello Sport the exact opposite happens, showing

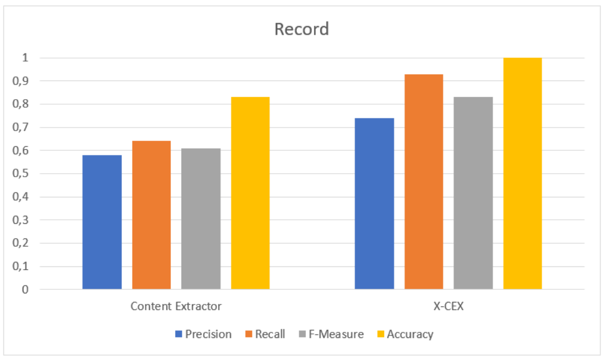

Figure 6: Comparison between X-CEX and Content Extractor for Record

that the X-CEX algorithm ends up worsening the results for this template, although it still improves Precision and Accuracy as can be seen in Figure 7.

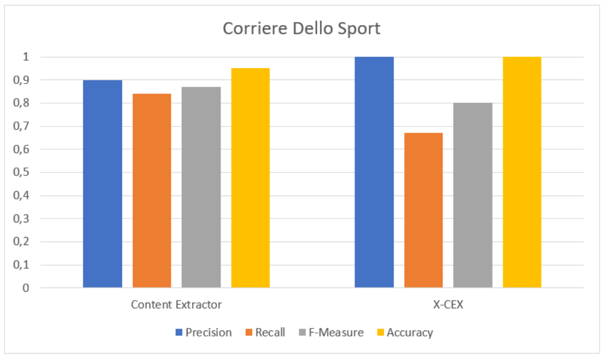

Figure 7: Comparison between X-CEX and Content Extractor for Corriere dello Sport

In general, there is mostly an increase in the level of Precision and Recall results, we can also observe that, there is a clear decrease in the number of blocks extracted in most of the Web Sites. This number is a reflection of the block extraction phase.

We still can observe that the Hamming distance manages to obtain a huge reduction in the execution time compared to the X-CEX algorithm, maintaining the results for Precision, Recall, F-Measure and Accuracy. I was surprised with the the results obtained, only being able to explain them perhaps by the factor of the Hamming distance to take into account only the number of positions that do not correspond in the two 'strings' provided. In this way, I believe that calculating a similarity of content using the Cosine of binary vectors for the terms used in the blocks, turns out to be more exhausting because it has to go through all the words in advance, unlike the Hamming distance.

5. Conclusions

5.1. Discussion

To finish I am very happy with the proposed solution, The algorithm implemented and proposed by me to extract informative content from news web pages, was the X-CEX algorithm, and we can conclude that this algorithm is not affected by the news languages, and it can present good results for several languages, which was an important component to be evaluated.

Now, at the end, I can answer the questions that arise, the components that most influence this extraction are, for all the work developed and analyzed, the initial filter/extraction of the blocks, that is, the beginning phase where each page is divided into several sections filtering the blocks that went to the comparison phase. And then, but not least, the system used to calculate the similarity between two blocks. This is one of the most prevalent phases of the process that has been optimized as much as possible in order to obtain the best possible results, the system that was used in this work it was a system of importance for each of the similarities (Structure/Style and Content) unlike the Content Extractor that use a system of joining both similarities together.

In conclusion, I was able to understand that this area is analyzed for a lot of people and because of that, there are already many alternatives to the solution of this problem, however, I also have to mention that there seem to be too many options little explored in my opinion, I think that the existing solutions should be more detailed and analyzed, there should be a greater focus on the existing methods and there should not be so many different options.

5.2. Limitations & Future Work

The biggest limitation of my algorithm is, being limited to analyzing page templates, it only works for a set of pages of the same class and it stops working correctly when the number of pages provided is very low, it loses efficiency for not being able to identify the Web Site template.

Finally, another of the great limitations of this algorithm is not being able to extract the content blocks that are outside the tags used for the block extraction phase. This is a major limitation, because when Web Sites change radically in structure, this algorithm fails to obtain good results. For this reason, in some Web Sites, the blocks of titles and sub-titles, for example, can be left out wrongly, the same thing happens for news images or new external publications to the site.

As a future work for this algorithm, I recommend exploring the block extraction phase well, this is the biggest gap in the algorithm because it is too restricted to certain templates. Now, this extraction phase uses a system of list of ordered tags, however, I believe that this is not the best method, ending up removing many relevant blocks from the comparison phase, some criteria must be used in conjunction with this. It is important to automatically remove certain areas from the pages, however, in the main sections other methods should be used, for example, taking the text or other components of those sections.

References

[1] S. S. Bhamare and B. V. Pawar. Survey on Web Page Noise Cleaning for Web Mining. *Imternational Journal of Computer Science and Information Technology*, 4(6):766–770, 2013.

[2] S. Debnath, P. Mitra, N. Pal, and C. L. Giles. Automatic identification of informative sections of Web pages. *IEEE Transactions on Knowledge and Data Engineering*, 17(9):1233–1246, 2005.

[3] A. Finn, N. Kushmerick, and B. Smyth. Fact or Fiction: Content Classification for Digital Libraries. In A. F. Smeaton and J. Callan, editors, *Proceedings of the Second {DELOS} Network of Excellence Workshop on Personalisation and Recommender Systems in Digital Libraries, {DELOS} 2001, Dublin, Ireland, June 18-20, 2001*, volume 01/W03 of *{ERCIM} Workshop Proceedings*. ERCIM, 2001.

[4] D. Freitag. Machine learning for information extraction in informal domains. *Machine Learning*, 39(2):169–202, 2000.

[5] P. U. Jadhav. Extracting Information from website using Site Style Tree. *International Journal on Recent and Innovation Trends in Computing and Communication*, 2(6):1654–1656, 2014.

[6] S. H. Lin and J. M. Ho. Discovering informative content blocks from Web documents. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 588–593, 2002.

[7] M. Marathe, S. H. Patil, G. V. Garje, and M. S. Bewoor. Extracting Content Blocks from Web Pages. *International Journal*, 2(4):62–64, 2009.

[8] D. Pinto, M. Branstein, R. Coleman, W. B. Croft, M. King, W. Li, and X. Wei. QuASM: A system for question answering using semi-structured data. In *Proceedings of the ACM International Conference on Digital Libraries*, 2002.

[9] M. F. Porter. An algorithm for suffix stripping, 1980.

[10] A. K. Rajashri Shinde, Ashwini Bolli. Methods For Extracting Content Blocks From Web Pages. *International Journal of Engineering Research & Technology (IJERT)*, 2(4):1–10, 2013.

[11] S. Vijayarani and K. Geethanjali. Web Page Noise Removal - A Survey. *International Journal of Scientific Research in Scienceand Technology*, 3(7):172–181, 2017.

[12] L. Yi, B. Liu, and X. Li. Eliminating noisy information in Web pages for data mining. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003.