

Improving the Boilerpipe Algorithm for Boilerplate Removal in News Articles Using HTML Tree Structure

Francisco Viveros-Jiménez¹, Miguel A. Sanchez-Perez¹, Helena Gómez-Adorno¹,
Juan-Pablo Posadas-Durán², Grigori Sidorov¹, Alexander Gelbukh¹

¹ Instituto Politécnico Nacional (IPN),
Centro de Investigación en Computación (CIC),
Mexico

² Instituto Politécnico Nacional (IPN),
Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad Zacatenco (ESIME-Zacatenco),
Mexico

{pacovj, masp1988}@hotmail.com, helenadorno@gmail.com,
jposadas@esimez.mx, sidorov@cic.ipn.mx, gelbukh@gelbukh.com

Abstract. It is well-known that the lack of quality data is a major problem for information retrieval engines. Web articles are flooded with non-relevant data such as advertising and related links. Moreover, some of these ads are loaded in a randomized way every time you hit a page, so the HTML document will be different and hashing of the content will be not possible. Therefore, we need to filter the non-relevant text of documents. The automatic extraction of relevant text in on-line text (news articles, etc.), is not a trivial task. There are many algorithms for this purpose described in the literature. One of the most popular ones is Boilerpipe and its performance is one of the best. In this paper, we present a method, which improves the precision of the Boilerpipe algorithm using the HTML tree for selection of the relevant content. Our filter greatly increases precision (at least 15%), at the cost of some recall, resulting in an overall F1-measure improvement (around 5%). We make the experiments for the news articles using our own corpus of 2,400 news in Spanish and 1,000 in English.

Keywords. Boilerplate removal, news extraction, HTML tree structure, Boilerpipe.

1 Introduction

The Web has become one of the most important sources of information with an extensive and diverse audience. Nowadays, news often get

published on the Web before appearing in television or newspapers. Thus, it is natural that many companies have shown interest in storing copies of particular content present on the Web [24, 11, 17]. Still, while storing these articles within an information retrieval engine, it is necessary to avoid storing duplicates. Moreover, articles are flooded with non-relevant content that changes randomly between downloads (mostly, it is advertising). It is well-known that poor data is one of the main difficulties in further processing, so data cleaning is usually performed [5, 7, 6]. According to [23]:

“Data quality problems are present in single data collections, such as files and databases, e.g., due to misspellings during data entry, missing information or other invalid data. When multiple data sources need to be integrated, e.g., in data warehouses, federated database systems or global web-based information systems, the need for data cleaning increases significantly.”

This paper focuses on the task of extracting the relevant content from a web page and removing the irrelevant content (a.k.a. boilerplate removal). Boilerplate removal is difficult because:

(1) Web articles usually have many non-relevant content, (2) Non-relevant content could be placed everywhere in the structure of the document (even in the middle of the text), (3) There are no style, design or structure rules for its detection, and (4) Non-relevant content can talk about topics related to the relevant fragments. Therefore, purely HTML-based or purely text-based approaches do not have perfect results.

In our particular case, we were working together with a media-related company, which has thousands of people manually extracting news from TV, radio, newspapers and the Web. They extract news in English and Spanish. We were given the task of helping to reduce the amount of time each worker have to deal with any particular website, so they can be assigned other tasks. Therefore, we need a method with high precision, but we can sacrifice some recall.

There are many approaches in the state-of-the-art for the task of boilerplate removal. We selected one of the most popular ones: Boilerpipe [19]. Boilerpipe has good all around performance, but it allows a considerable amount of non-relevant content (i.e., it is more oriented on recall than on precision). Our main goal was to keep – or improve – the algorithm performance focusing on its precision: changing the final performance into a high precision / decent recall.

Boilerpipe considers the task of boilerplate removal as a binary classification task: “classify every text node in the HTML tree as relevant or non-relevant”. It uses wide amount of text-based features and structure-based features. Boilerpipe considers that any text node can be relevant. Its main problem is that it selects a considerable amount of non-relevant content. We discovered that this problem can be mitigated by using the HTML tree in cleverer way to prune some non-relevant text selected by the original Boilerpipe. Our testing has shown that our improvement has better precision and F-measure than Boilerpipe (at the cost of some recall). We also present a comparison with another state-of-the-art method: Justext [22]. Justext is a method that mainly uses the HTML tree.

It picks the best group using a score generated through detailed calculations of features extracted from text and the tree itself.

2 Related Work

There are several methods for boilerplate removal and lots of ways of grouping them. We consider that the best way to classify them is: domain-dependent, visual-based, and HTML-based. A domain-dependent method usually needs a considerable amount of resources per site, such as media files, style files or a sufficiently large number of pages from the same website. There are many approaches that fall into this category [10, 25, 9, 21, 8, 4]. It is time consuming task to prepare these resources for each Web site and the methods depend on the availability of data, so we did not use these methods in our work.

A visual-based method uses style and visualization features. Therefore, they need to render the HTML file, thus requiring the additional download of style files, script files and possibly pictures. They use visual page segmentation as the core process [3]. We can find an example of visual-based method, say, in [12]. We did not use visual-based methods, because they required a considerable amount of extra downloads, while most websites of newspapers have limited download rates. An HTML-based method only requires the HTML file. They can be used for any website at any time. There are many HTML based approaches [20, 14, 1, 22, 15, 16]. Approximately half of these methods work classifying every text node as relevant or non-relevant. The other methods work by selecting the best node using the HTML tree and some heuristics. HTML-based methods have low additional requirements and are robust, so we developed a method, which falls in this category.

3 Description of the Boilerpipe Method

Boilerpipe is an algorithm that processes an HTML file and returns the HTML tree corresponding to the main content of the file.

It uses a binary classifier for deciding if a text node is relevant or not. In this case, every text node is classified independently. The main purpose of the Boilerpipe algorithm is to be simple and domain independent. The main contribution of the method is the feature set used for classification. The method uses structural features, shallow text features, densiometric features and the frequency of specific text fragments in the training corpus. We describe these features below.

Structural features maintain the inherent semantics of the HTML tags. In particular, the following features are used: headline tags ($h1$, $h2$, ...), paragraph tags (p), division tags (div), and anchor text tags (a).

Shallow text features treat text at the functional level rather than at topic-related level. These features are not based on the bag of words or n -gram models. Instead, they use values that are both domain and language independent such as average word length, average sentence length, absolute number of words, absolute and relative position of a text block in the document. Some additional features are defined to deal with word capitalization, date/time-related tokens, special navigational characters and link density of a section.

Densiometric features deal with the distribution of text in the HTML tree. Boilerpipe uses text density [18], which is the measure for estimation of how much text is contained within a block with respect to conventional language measures.

The experiments using several classifiers confirm that all these features together yield the best performance.

4 Proposed Method for Content Filtering

We have used Boilerpipe extensively to crawl some newspapers websites for research purposes. In our experience, Boilerpipe usually retrieves a considerable amount of irrelevant text. Our hypothesis is that these irrelevant text can be discarded by using the HTML tree. We consider that good content corresponds to a single node in the HTML tree.

According to [22], blocks tend to create clusters (meaning that the relevant content tags are usually near each other in the HTML tree). But how can we identify the relevant HTML node? Also, how can we discard non-relevant content inside that important node?

We propose quite straight-forward idea: use Boilerpipe – that has good recall values – for selection of a set containing mostly relevant content and then filter out bad content by using the HTML tree. Our filtering procedure is as follows:

1. Identify if a text node is a paragraph or not. A text node is a paragraph when it has one of the following tag names: *div*, *table*, *ul*, *ol*, *p*, *section*, *article*, *h1*, *h2*, *h3*, *h4*, *h5*, *h6*, *header* and *body*. A paragraph should not contain other paragraphs. In this case we use the closest child satisfying this condition instead.
2. Generate an ancestor list for all the selected paragraphs.
3. Group all the paragraphs having the same n -th parent.
4. Select the group having the most amount of text (discard everything else).

This filtering procedure is simple, but efficient. The next sections analyze the performance, advantages and flaws of our improvement. We also tested various depth values: up to 5, see the section “Experimental results”, obtaining the best results with grandparents (depth=2).

5 Experimental Settings

First, we tested our approach using the CleanEval [2], corpus and obtained results a little bit worse than the state-of-the-art (see Section 7). But note that our approach is designed specifically for complete news articles, while CleanEval corpus has many texts of different genres: blog posts, comments, forums and section pages, etc. Besides, it considers comments as valid documents.

Thus, we created our own corpus in order to evaluate our approach in our specific niche: news articles.

It is justified by the practical purpose of work, when our business partner only is interested in news articles, which is also a challenging task. We used the texts written in English and Spanish languages. Our corpus is composed by 2,400 news taken from 14 websites in Spanish and 1,000 news extracted from 10 websites in English. All of the selected websites correspond to major news agencies from 7 different countries. We retrieved equal number of news from each site. The relevant content from each article was manually extracted by 4 annotators in order to build the gold standard. Every article was reviewed by 2 annotators. Our corpus is freely available¹.

We measured the performance of the approach using precision, recall and F1-measure. These measures were calculated as follows:

$$\text{precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}, \quad (1)$$

$$\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}, \quad (2)$$

$$F1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}, \quad (3)$$

where a positive value is any character that belongs to the gold standard content and a negative value is any character that should have been discarded.

6 Empirical Analysis of the Corpus

Usually, web pages have a lot of non-related content. Some researchers estimated in 2005 that around 50% of the content is irrelevant [13]. They also said that the amount of irrelevant content was growing yearly. We observed that nowadays it has grown greater than 60%: 73% in English and 63% in Spanish. Note that in general English articles are bigger and have more irrelevant content. Therefore, they are harder to clean than Spanish articles.

Examples of the non-relevant content that was successfully removed are presented in Fig. 1 (related content) and Fig. 2 (content with advertizing).

¹<http://www.cic.ipn.mx/~sidorov/>.

Table 1. Comparison of the boilerplate removal algorithms for English and Spanish

Approach	Spanish			English		
	P	R	F	P	R	F
Boilerpipe	0.83	0.98	0.90	0.76	0.92	0.83
Justext	0.86	0.93	0.89	0.85	0.84	0.84
Ours	0.97	0.93	0.95	0.95	0.84	0.89

Table 2. Performance changes caused by using different ancestors (values 1 to 5 indicate the tree distance from the node to its ancestor, i.e., 1=father, 2=grandfather, etc.)

	1	2	3	4	5
Spanish					
P	0.97	0.97	0.91	0.90	0.89
R	0.92	0.93	0.95	0.95	0.95
F	0.94	0.95	0.92	0.92	0.91
English					
P	0.96	0.95	0.90	0.87	0.85
R	0.82	0.84	0.88	0.88	0.89
F	0.89	0.89	0.89	0.87	0.87

7 Experimental Results

We tested three approaches: Boilerpipe, Justext [22] and our content filtering approach based on Boilerpipe. Results in Table 1 confirm that our filtering approach with Boilerpipe significantly increases the precision and F-measure values of Boilerpipe (at the cost of some recall). Our algorithm also has better F-measure value than the Justext approach (which is one of the best approaches for boilerplate removal [22]). We confirmed that all of our differences were statistically significant through a Mann-Whitney U test using a confidence of 99%.

We tested the usage of other types of ancestors instead of grandparents. Results in Table 2 confirm that using grandparents is the best choice. However, using parents is also a competitive option.

We performed a manual check of 200 articles to analyze the behavior of our approach. We observed that the content usually forms groups in the HTML tree. The most common groups were: (1) Main content; (2) Navigational links; (3) Advertising; and (4) Links to the related content (that are usually grouped with article excerpts and pictures). Our filter behaves in the following way:



Fig. 1. Related content group

prison at Guantánamo, saying it had not proven to be an inspiration for militants.

Get the Morning Briefing by Email

What you need to know to start your day, delivered to your inbox Monday through Friday.

☒ Receive occasional updates and special offers for The New York Times's products and services.

☐ No soy un robot
 

SEE SAMPLE | PRIVACY POLICY

General Kelly also questioned the Obama administration's [plans to open all combat jobs](#) to women, saying the military would have to lower its physical standards to bring women into some roles.

"There will be great pressure, whether it's 12 months from now, four years from now, because the question will be asked whether we've let women into these other roles, why aren't they staying in those

other roles?" he said to reporters in January, shortly before his retirement.

Fig. 2. Advertising

1. It correctly removes almost all of the navigational links (like "Advertisement", "Share this:" or "Order Reprints — Today's Paper — Subscribe").
2. It correctly removes almost all the related content excerpts and links.
3. It correctly removes a great amount of advertising. Only some links that are carefully placed as siblings of the main content pass the filter (the precision is above 95%, which is acceptable for our clients).
4. It wrongly discards a considerable amount of summaries and quotes.
5. It wrongly selects some paragraphs regarding content such as "as A version of this article appears in print on October 14, 2016, on page A25 of the New York edition with the headline: Trepidation and Outrage at City College in Wake of President's Abrupt Exit."
6. It cannot recover anything that was wrongly discarded by Boilerpipe.

Our filtering does not have trouble handling tables, because table tags are considered paragraphs and are usually placed as siblings of the other content nodes. However, Boilerpipe frequently discards tables.

As we mentioned before, we also tested our method in the CleanEval test set and got results that are a little bit worse than state-of-the-art. Our method obtained: $P = 0.96$, $R = 0.66$, $F = 0.78$, while BoilerPipe obtained $P = 0.95$, $R = 0.74$, $F = 0.83$. So, our method gained a little bit of precision in exchange of a certain recall loss. The main reasons behind this behavior are:

- CleanEval has more diverse content (only around 60% are regular articles). Mailing lists, blog covers, forums and section pages cause problems to our filtering strategy. It is a common practice in those types of pages to have its relevant content distributed in distant tree branches. Our method only selects a single branch and therefore discards a great amount of relevant content. Our filter is good for regular articles only.
- CleanEval considers comments in articles as the relevant content. We remove these comments in our test set because there are not relevant for our commercial objective. Our filtering strategy removes all the comments.

8 Conclusions and Future Work

This paper presents a novel method for content filtering (boilerplate removal) for specific type of texts: new articles. It improves the results of the popular boilerplate removal algorithm: Boilerpipe. Our filter greatly increases precision (at least 15%) at the cost of some recall, resulting in an overall F1-measure improvement (around 5%). Further research will be conducted to implement our method as the independent procedure.

Acknowledgements

This work was partially supported by the Mexican Government (CONACYT project 240844, SNI, COFAA-IPN, SIP-IPN 20171344, 20171813, 20172008).

References

1. Bar-Yossef, Z. & Rajagopalan, S. (2002). Template detection via data mining and its applications. *Proceedings of the Eleventh International Conference on World Wide Web, WWW '02*, pp. 580–591.
2. Baroni, M., Chantree, F., Kilgarriff, A., & Sharoff, S. (2008). CleanEval: a competition for cleaning web pages. *Proceedings of the Sixth International Conference on Language Resources and Evaluation, LREC '08*.
3. Cai, D., Yu, S., Wen, J.-R., & Ma, W.-Y. (2003). Extracting content structure for web pages based on visual representation. *Proceedings of the Fifth Asia-Pacific Web Conference on Web Technologies and Applications, APWeb '03*, pp. 406–417.
4. Chakrabarti, D., Kumar, R., & Punera, K. (2007). Page-level template detection via isotonic smoothing. *Proceedings of the Sixteenth International Conference on World Wide Web, WWW '07*, pp. 61–70.

5. Christen, P. (2012). A survey of indexing techniques for scalable record linkage and deduplication. *IEEE transactions on knowledge and data engineering*, Vol. 24, No. 9, pp. 1537–1555.
6. Churches, T., Christen, P., Lim, K., & Zhu, J. X. (2002). Preparation of name and address data for record linkage using hidden markov models. *BMC Medical Informatics and Decision Making*, Vol. 2, No. 1, pp. 9.
7. Clark, D. (2004). Practical introduction to record linkage for injury research. *Injury Prevention*, Vol. 10, No. 3, pp. 186–191.
8. Debnath, S., Mitra, P., Pal, N., & Giles, C. L. (2005). Automatic identification of informative sections of web pages. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17, No. 9, pp. 1233–1246.
9. Endr dy, I. & Nov k, A. (2013). More effective Boilerplate removal—the GoldMiner algorithm. *Polibits*, Vol. 48, pp. 79–83.
10. Evert, S. (2008). A lightweight and efficient tool for cleaning web pages. *Proceedings of the Sixth International Conference on Language Resources and Evaluation*, LREC '08.
11. Ferrara, E., Meo, P. D., Fiumara, G., & Baumgartner, R. (2014). Web data extraction, applications and techniques: A survey. *Knowledge-Based Systems*, Vol. 70, pp. 301–323.
12. Gao, W. & Abou-Assaleh, T. (2007). Genieknows web page cleaning system. *Building and Exploring Web Corpora: Proceedings of the Third Web as Corpus Workshop, incorporating Cleaneval*, pp. 135.
13. Gibson, D., Punera, K., & Tomkins, A. (2005). The volume and evolution of web page templates. *Special Interest Tracks and Posters of the Fourteenth International Conference on World Wide Web*, WWW '05, pp. 830–839.
14. Gibson, J., Wellner, B., & Lubar, S. (2007). Adaptive web-page content identification. *Proceedings of the Ninth Annual ACM International Workshop on Web Information and Data Management*, WIDM '07, pp. 105–112.
15. Girardi, C. (2007). Htmcleaner: Extracting the relevant text from the web pages. *Proceedings of the Third Web as Corpus Workshop*, WAC '07, pp. 15–16.
16. Hofmann, K. & Weerkamp, W. (2007). Web Corpus Cleaning using Content and Structure. *Building and Exploring Web Corpora: Proceedings of the Third Web as Corpus Workshop*, WAC '07, pp. 145–154.
17. Kilgarri f, A., Rychly, P., Smrz, P., & Tugwell, D. (2004). The sketch engine. *Proceedings of EURALEX*, pp. 105–116.
18. Kohlsch t ter, C. (2009). A densitometric analysis of web template content. *Proceedings of the Eighteenth International Conference on World Wide Web*, WWW '09, pp. 1165–1166.
19. Kohlsch t ter, C., Fankhauser, P., & Nejdl, W. (2010). Boilerplate detection using shallow text features. *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, pp. 441–450.
20. Marek, M., Pecina, P., & Spousta, M. (2007). Web page cleaning with conditional random fields. *Building and Exploring Web Corpora: Proceedings of the Third Web as Corpus Workshop, incorporating Cleaneval*, pp. 155.
21. Pasternack, J. & Roth, D. (2009). Extracting article text from the web with maximum subsequence segmentation. *Proceedings of the Eighteenth International Conference on World Wide Web*, WWW '09, pp. 971–980.
22. Pomik lek, J. (2011). *Removing Boilerplate and Duplicate Content from Web Corpora*. Doctoral theses, dissertations, Masaryk University, Faculty of Informatics, Brno.
23. Rahm, E. & Do, H. H. (2000). Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.*, Vol. 23, No. 4, pp. 3–13.
24. Sch fer, R. & Bildhauer, F. (2012). Building large corpora from the web using a new efficient tool chain. *Proceedings of the Eight International Conference on Language Resources and Evaluation*, LREC '12, pp. 486–493.
25. Yi, L., Liu, B., & Li, X. (2003). Eliminating noisy information in web pages for data mining. *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pp. 296–305.

Article received on 09/10/2017; accepted on 12/01/2018.
Corresponding author is Grigori Sidorov.