Removing Noise Content from Online News Articles

Jayendra Barua
Indian Institute of Technology,
Roorkee, Uttarakhand
India 247667
jbarudec@iitr.ac.in

Dhaval Patel
Indian Institute of Technology,
Roorkee, Uttarakhand
India 247667
patelfec@iitr.ac.in

Ankur Kumar Agrawal
Indian Institute of Technology,
Roorkee, Uttarakhand
India 247667
ank70pec@iitr.ac.in

ABSTRACT

A typical news web page consists of news articles. Along with the news article content tags, it also contains tags of navigation links, privacy & copyright information and advertisements. These tags are called as noise tags. Given an online news article in html form, existing works extract articles by discovering informative tags using various heuristic techniques. In this paper, we follow an alternate approach that removes noise tags from the web page. In particular, we define two types of noise contents, namely *static noise content* and *dynamic noise content*, to be removed from a web page. Static noise content is the content which is present in all the article web pages of a same news website, whereas, the dynamic noise contents are advertisements and irrelevant hyperlinks which keep changing from one article web page to another web page. We present a two-stage approach for identification of static and dynamic noise content. After identifying tags with *noise content*, we remove these tags from the news webpage and thus only tags with article content will be left out. Experimental studies is done on news web pages extracted from 11 different news websites. Comparison with three well known open source content extraction techniques our approach suggest around 6% improvement in recall value with fair F1-score and precision value compared to best performing content extraction technique.

1. INTRODUCTION

Along with increase in publication of online news articles, many news aggregators, news recommendation and search systems such as Google News, Yahoo News etc. has been evolved which crawls various news websites and then extract the news content from the crawled news web pages. However, news content extraction from a web page is not a simple task because, along with the article content, news web pages also contain "*Noise*" in the form of irrelevant hyperlinks, advertisements, copyright information, etc. Yi et al. in [1] have also shown that noise can seriously harm the accuracy of various data mining algorithms that are performed on news article content. Given online news article in a form of an html form, existing works extract articles by discovering informative tags using various heuristic techniques [2] [4] [7]. In particular, these heuristic techniques calculate various statistics for each tag present in the web page and then identify the informative tags. Since, news aggregators crawl new websites multiple times for durable archival of contents, many news articles from the same news website are accumulated. In such scenario, it is possible to learn how a website publishes its own news articles. In particular, by analyzing a set of news articles published by a single news website, we can easily distinguish the noisy and informative content. Existing work [1] and [5] also uses the multiple news articles for efficient content extraction, but none of these technique leverages static and dynamic nature of the noise content in web pages.

The 20th International Conference on Management of Data (COMAD), 17th-19th Dec, 2014 at Hyderabad, India.

Given a set of web pages *W* that are sampled from a single news web site N, we develop a document object model (DOM) tree based solution to remove the noise contents of each web page in *W*. Each web page in *W* is parsed using HTML parser and is represented as a DOM tree. Figure 1 shows a DOM tree representation of a news web page (some details omitted) where each node is an HTML tag. We have some observations based on DOM tree as follows: (1) Some nodes in DOM tree have same content in all the news web pages of *W*, for example Home, name of news website, copyright, print, follow us, contact us and many more. These content are not relevant to article text. We called these types of nodes as *Static Noise Tags*. These nodes may be either text nodes or hyperlink nodes. (2) Other noise nodes in the DOM tree are hyperlinks. Here some nodes are relevant to the article text content, but most of the nodes are irrelevant. Many of the irrelevant hyperlinks are generally listed under nodes such as Recent articles, Mostly Read articles, related news, etc. We call these types of nodes as *Dynamic Noise Tags*. We term these tags as "Dynamic" because information under these tags is not static across all news web pages *W*.

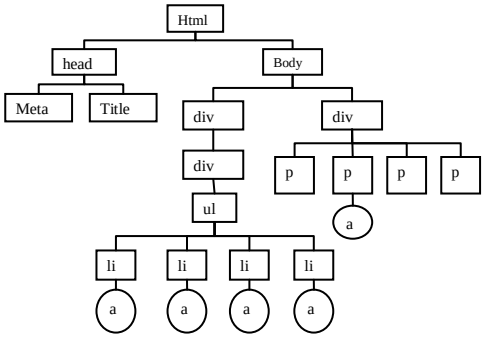

Figure 1: DOM tree of a news web page

In this paper, we use a two step approach, namely StaDyNoT, to remove *Static and Dynamic Noise Tags*. In the first step, we mark *Static Noise Tags* by using set of web pages extracted from the same website. In the second step, we mark *Dynamic noise Tags* by applying the common ancestor technique on hyperlink nodes of DOM tree of the given news web page. Finally, we remove all noise marked tags and thus only tags with article content will be left on the web page.

Moreover, existing content extraction techniques output unformatted text from input web page. These methods cannot be used to extract structured data from the web pages since there output looses text formatting such as font-size, bold and italic text, etc. In contrast, instead of unformatted text, our technique outputs cleaned HTML web pages which retains text formatting. If needed, only unformatted text can also be easily extracted from these HTML web pages**.** Based on exprimental analysis, StaDyNoT achieves an overall average score of 88%, 96% and 91% for Precision, Recall and F1 respectively, which is remarkable in comparison with existing NReadability[1], Boilerplate[8] and *jusText*[3], content extraction techniques.

2. RELATED WORK

Retrieval of noiseless news content has attracted many researchers[1][2][3][4][5][6][7][8]. In order to retrieve noiseless news content, majority of the existing approaches searches for the article content in the web page and extract it. Compared to this, our method searches for noise content and remove it.

Yi et al. in [1] introduces a Site Style tree to capture common-styles element tags in web pages of a website. Then by identifying common-style they remove noise element tags from web pages. But the noise content on news web pages may not follow the common style always. Fan et al. in [2] proposes the detection of "print-link" on the web pages to extract article content. But many variations of the print-link representations makes print-link detection a non-trivial problem. A set of rules are used in *jusText* [3] that classify text blocks as "good" or "bad". The classification is done by applying context-free rules followed by context-aware rules. The work in [4] uses text density to extract the informative content. Sluban et al. in [5] introduces URL tree created using of stream of HTML web pages to extract the main content. In [9] a heuristic approach based on text-density and visual importance features of a web page is proposed by Song et al. for extraction of article content. Bhu et al. in [10] proposed a heuristic based approach which uses "fuzzy association rules" and "sliding window" to extract main text content from web pages. Peters et al. in [6] introduces tag attribute values as new block feature to improve article extraction. ECON method by Guo et al. in [7], works on principle that informative content contains more number of punctuation marks than noise content. Kohlschutter et al. in [8] proposes the detection of noise (termed as boilerplate) on the basis of shallow text features, structural features and densito-metric features of text blocks in a web page. NReadability[1] is a tool, which removes clutter from HTML pages by applying several heuristics on DOM tree, based on textual and structural features of web page.

In comparison to the existing methods which are mostly heuristic based or supervised, we propose a simple, unsupervised and efficient approach which leverages static and dynamic nature of noise content for removing noise from news web pages.

3. OUR APPROACH: StaDyNoT

As discussed in the Introduction section, if we can remove both S*tatic Noise Tags* and *Dynamic Noise Tags* from news web page then only article text content will remain in the web page. Our approach works in 2 stages: 1) Marking of *Static Noise Tags* 2) Marking of *Dynamic Noise Tags.* Note that, in each stage we only mark the noise tags, but we do not remove noise tags. This is because removal of certain nodes may distort the DOM tree structure and we may not be able to process the tree in subsequent stage.

3.1 Mark Static Noise Tags

This stage utilizes a set of neighbor article web pages to mark a static data that are present in a target web page. A target web page is a page from which we want to extract the articles. By the term "*neighbor article web pages*", we mean the set of news web pages belongs to the same news website and same category as of the target article news web page. For example if the target article news web page belongs to sports category of CNN, then other news web pages from the sports category of CNN will be neighbor article web pages. "neighbor article web pages" are naturally available in news aggregators systems, as they crawls multiple web pages of same websites, Thus we leverage availability of multiple news web pages belongs to same news website. In Figure 2, we explain the adopted steps in detail. The proposed algorithm works in two phases. In the first phase, we analyze the neighbor article web pages, learn static data and then mark the tags having static data as noise tags.

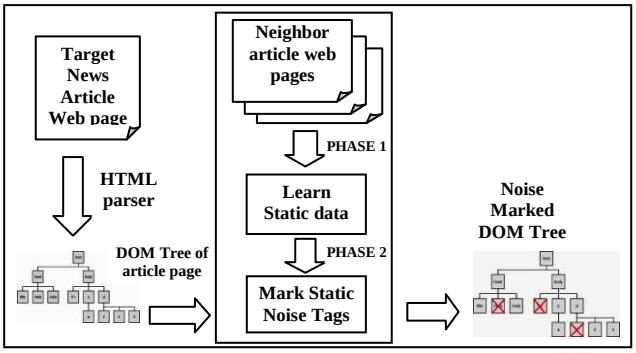

Figure 2: The process of marking *Static Noise Tags*

In particular, we traverse the DOM tree of every article web pages in neighbor article web pages. While traversal, we extract tag name, tag attribute and tag data of every node in a DOM tree and put them into a global hash table using key "tag-name : tag-attribute: tag_data" with support value 1. If similar key is already present in the global hash table, then we increment the value by 1. At the end, we prepared a hash table having tag information along with their support count. Finally, we iterate each key-value pair in the hash table and output a tag having value equals to the number of neighbor pages as a *Static Noise Tag*. Table 1 lists examples of some of the content of *Static Noise Tags* identified for the sport category of CNN using 10 neighbor articles.

Table 1: Example of *Static Noise Tags*

Subset of *"Static Noise Tags" content* of CNN Sports Category
Terms of service | Privacy guidelines | Ad choices | Advertise with us | License our content | About us | Contact us | Work for us | Help:
EDITION: INTERNATIONAL U.S. MÉXICO ARABIC TV: CNNi CNN en Español Set edition preference Sign up Log in: Most Popular:
Tools & Widgets | RSS | Podcasts | Blogs | CNN Mobile | My Profile | E-mail Alerts | CNN Shop | Site map | CNN Partner Hotels:
Comments » SHARE THIS Print Email More sharing Reddit StumbleUpon
© 2013 Cable News Network. Turner Broadcasting System, Inc. All Rights Reserved.:

Once, *Static Noise Tags* are identified, we traverse DOM tree of the target web page and mark the node as a noise tag if the node matches with any of the identified Static Noise Tag. For marking a tag as noise tag we add a special attribute "NOISE" to the tag.

[1] https://github.com/marek-stoj/NReadability

3.2 Mark Dynamic Noise Tags

Now, we have a DOM tree of the target web page in which *Static Noise Tags* are marked. However, we noticed that there are many noise nodes in DOM tree that have dynamic contents. For example, the node related to "Recent News" section contains hyperlinks that occur together under a <ul> tag. Such tags have high percentage of hyperlink content. We call such type of tags as *Dynamic Noise Tags*. In Figure 3, we present an example of such tag. In this example, "ul" node (in dotted ellipse) is identified as *Dynamic Noise Tag*. Some hyperlinks may also occur under article text nodes, but percentage of hyperlink content is quite low in such article text nodes. Our aim is to identify and mark *Dynamic Noise Tags* in the DOM tree of the target web page.

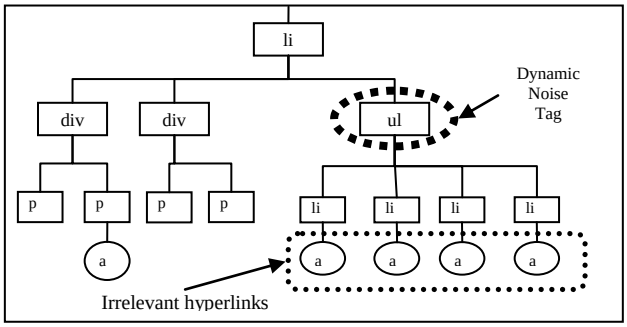

Figure 3: Illustrating irrelevant hyperlinks under *Dynamic Noise Tag* in part of DOM Tree

We apply Least Common Ancestor (LCA) on the output DOM tree of Stage 1 to find the *Dynamic Noise Tags*. Given a DOM tree, we represent each node with tag "a" (i.e., hyper-link tag) by a path-string. A path-string of a node *n* is a path from root to the node *n* in DOM tree along with the position information of each node on the path. In our case, position information of a node is obtained with respect to its parent. For example, consider a path-string for node with tag "p":

0(html)-1(body)-12(div)-0(div)-3(p)

This path string is for representing a "p" tag in the DOM tree since it appears last in the string. Value "3" just before "p" in the string indicates that node "p" is the 4th child of "div" tag which is second last in the string. This "div" tag is 1st child of its parent "div" tag which is 13th child of "body" tag and body tag is 2nd child of "html" tag. Here "html" tag is root tag. The path-strings of tag <a> in Figure 3 is Illustrated in Figure 4. Note that, the path-strings shown in Figure 4 are complete however only a partial DOM tree is shown in Figure 3. As we can see from the Figure 3 and 4, the *Dynamic Noise Tags* are least common ancestor of hyperlink nodes. (html)-1(body)-12(div)-0(div)-3(footer)-0(div)-0(div)-4(ul)-0(li)-2(ul): is the Candidate *Dynamic Noise Tag*.

Thus, after obtaining the set of path-strings for each anchor (<a>)node, we identify least common ancestors of discovered path-strings. These least common ancestors are also nodes in a DOM tree and marked as a C*andidate Dynamic noise Tag*. However, there may be hyperlink nodes inside article text nodes. So some article text node may be identified as *Dynamic Noise Tags* if we follow only LCA method. To avoid marking nodes related to informative content as a *Dynamic Noise Tag*, we further apply two simple filtering strategies for each node that are identified as a *Candidate Dynamic Noise Tag* by Least Common Ancestor method.

1) *Candidate Dynamic Noise Tag* with <p> tag cannot be a "*Dynamic Noise Tag*".

2) A *Candidate Dynamic Noise* is "*Dynamic noise Tag"*

- If the non-link text count (i.e. number of characters including spaces) appears in the node is less than the threshold α. **OR**
- If the link fraction of the node *n*, denoted as LF_n, is greater than β. Here, LF_n is defined as follow:

$$Link\ Fraction\ (LF_n) = \frac{Link\ text\ count\ of\ node\ n}{Total\ text\ count\ of\ node\ n}$$

Here α and β are threshold values for non-link text (no of characters) and Link Fraction respectively. We empirically determine the values α = 50 and β = 20.

This Stage may re-mark some of the nodes of Stage1, but this does not create any problems.

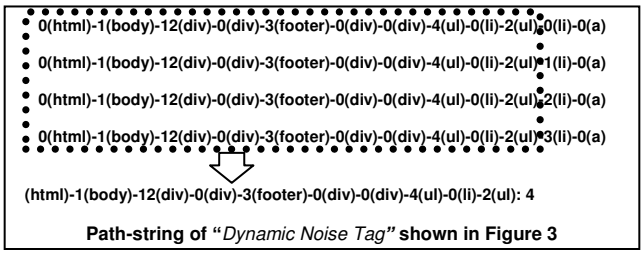

Figure 4: Illustration of identifying Dynamic noise nodes by applying LCA on path-strings

3.3 Post processing

After marking of *Static Noise Tags* and *Dynamic Noise Tags*, we are now having a noise marked DOM tree. In post processing we perform two operations: (1) **Remove Noise Marked Tags**: While marking of S*tatic and Dynamic Noise Tags*, DOM tree was marked by adding a new attribute "NOISE" to the tags. So these tags can be found by finding the tags having attribute "NOISE" and removed easily. (2) **Remove non-informative Formatting Tags**: There are some tags which never carry any informative content in them, because they are used for showing images, creating form elements, and formatting, styling and other purposes in html. So these tags are called as "Non-informative Formatting Tags". The list of such tags is shown below:

<img>,<link>,<script>,<style>,<noscript>,<embed>,<object>, <param>,<form>,<input>,<caption>,<meta>,<textarea>.

All the non-informative formatting tags should be removed along with Noise marked tags. Thus after post processing we will have the DOM tree with article text content only. The html web page of this DOM tree consists only article content.

4. EXPERIMENTS

We compare our technique "StaDyNot" with three techniques "Boilerplate[2]", JusText[3], and NReadability. For Boilerplate, we have used the "ArticleExtractor Instance" (as suggested by Boilerpipe API) for extracting the news articles. Our experiments are performed on 440 news article web pages having average corpus size 554 words, collected from 11 different news websites . To cover the variation of formatting and style of a news website, we have gathered about 10 pages from four categories, such as business, entertainment, politics and sports of each web site. Thus, in total we have total 11x4x10 web pages. We have manually prepared the ground truth for each web page in our dataset. While preparation of our ground truth dataset, along with the article text, we have also identified the title, author, date and time of

[2] https://code.google.com/p/boilerpipe/

[3] https://code.google.com/p/justext

Table 2: Comparative study of Article Extraction Techniques: StaDyNoT, NReadability, Boilerplate and JusText

News Source	StaDyNoT			NReadability			Boilerplate			JusText		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Anchorage daily	91.03%	98.80%	**94.68%**	90.82%	93.11%	91.69%	74.93%	95.70%	82.99%	75.05%	90.22%	80.97%
BostonHerald	90.09%	97.78%	**93.70%**	92.60%	93.09%	92.71%	89.37%	94.80%	91.81%	90.79%	87.45%	88.08%
BusinessWeek	92.70%	95.85%	**94.21%**	93.88%	93.81%	93.84%	90.82%	90.49%	90.64%	92.32%	88.29%	90.18%
DNA India	96.64%	99.63%	**98.09%**	87.42%	87.04%	87.16%	97.11%	98.87%	97.94%	92.60%	76.35%	77.85%
Chicago Tribune	90.82%	96.45%	**93.22%**	91.27%	92.57%	91.86%	86.67%	88.50%	87.29%	85.11%	82.34%	83.55%
CNN	72.93%	96.40%	81.82%	80.02%	91.65%	84.53%	73.62%	85.47%	78.67%	64.45%	80.70%	69.44%
DailyMirror (UK)	86.24%	99.19%	**91.26%**	77.89%	79.60%	78.09%	82.77%	84.90%	83.05%	86.48%	47.36%	49.57%
NDTV	88.34%	91.03%	89.56%	89.35%	87.42%	88.36%	89.67%	91.18%	90.36%	86.41%	73.59%	77.01%
BostonGlobe	90.70%	94.05%	**90.92%**	88.25%	91.12%	88.88%	83.89%	79.99%	81.30%	89.12%	87.85%	87.63%
Deccan Chronicle	76.99%	98.57%	83.71%	93.28%	96.34%	94.74%	84.38%	86.61%	85.20%	83.06%	91.23%	81.62%
DenverPost	90.60%	95.97%	**93.08%**	88.40%	90.04%	88.93%	89.50%	92.25%	90.72%	66.70%	82.07%	72.93%
Overall Result	**87.92%**	**96.70%**	**91.29%**	**88.47%**	**90.53%**	**89.16%**	**85.70%**	**89.89%**	**86.92%**	**82.92%**	**80.68%**	**78.59%**

publish as article content. Recall, our algorithm for marking *Static Noise Tags* uses neighbor article web pages for learning. While experimenting in this case, we use the news web pages of same category of a news website as "neighbor article web pages". For example, To mark S*tatic Noise Tags* in sports news article webpage of Boston herald, we will use 9 other pages from the sports category of Boston herald as "neighbor article web pages". Next, we extract articles from each news page and compare it with ground truth. We have utilized the standard evaluation measures, specifically, precision, recall and F1-scores were calculated by comparing the results of each method against the ground truth dataset. Note that, we have represented ground truth and extracted article as a bag of words that take into account the frequency of words without removing any stop-words. Table 2 shows the experiment results of our approach and other techniques.

For each news website, we have compared all the techniques on all the three measures precision (P), recall(R) and F1-score (F1) . Table 2 and Figure 5 shows summary of results. The results show average of precision, recall and F1-score values of all categories in each news website. Starting from poorest performing technique, *jusText* scored low score in all three measures compared to other methods. It indicates that simple heuristic based rules do not perform well on variety of data. Boilerplate technique is not extracting news title, author name and date-time of publish information from most of news web pages of DailyMirror(UK), AnchorageDaily, DeccanChronicle, BostonGlobe, and many other news web pages. *NReadability* is closest competitor of our approach and scored precision value equal to our StaDyNoT approach, but its recall value is 90%, which is low compared to our StaDyNoT approach. *NReadability* is not extracting the date-time and author information from AnchorageDaily, BostonGlobe, BusinessWeek and many other news websites, also many extracted articles missing some article text contents which affects its recall value. We can see in Figure 5 that our approach *StaDyNoT* performed well and scored highest F1-score, and recall value. Our approach has scored a significant 6% higher recall value compared to best performing *NReadability* technique. A good recall value helps web-search-engines for better indexing of extracted news articles.

5. CONCLUSION

In this paper we have proposed an effective approach for cleaning of news web pages by identifying the noise content on the basis of its "static" or "dynamic" nature among its neighbor web pages. We have observed that static noise content repeats itself in neighbor pages while most of the dynamic noise content accumulates under "*Dynamic Noise Tags*". We present a two stage algorithm for effective identification and removal of noise content from news web pages, while majority of other techniques emphasize on identification and extraction of informative content. Due to different strategy, our approach is able to achieve a recall value of 96% in experiments. It reflects that output of our approach contains all the article content in most of web pages with the overall precision of 88% which is quite fair.

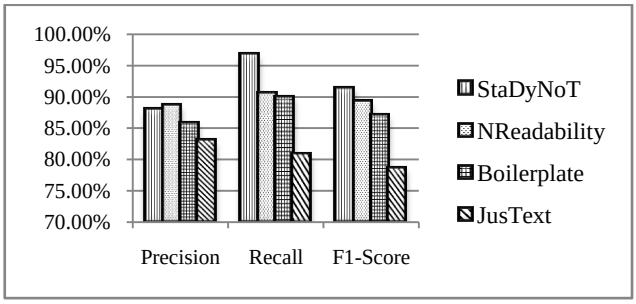

Figure 5: Result comparison among all the techniques

6. REFERENCES

[1] Lan Yi, Bing Liu, and Xiaoli Li. 2003. Eliminating noisy information in Web pages for data mining. *SIGKDD, ACM.*

[2] Jian Fan, Ping Luo, Suk Hwan Lim, Sam Liu, Parag Joshi, and Jerry Liu. 2011. Article clipper: a system for web article extraction. *SIGKDD, ACM.*

[3] J. Pomikalek. 2011. Removing Boilerplate and Duplicate Content from Web Corpora. PhD thesis, Faculty of Informatics, Masaryk University, Brno, Czech Republic,

[4] Fei Sun, Dandan Song, and Lejian Liao. 2011. DOM based content extraction via text density. *SIGIR, ACM* .

[5] Borut Sluban and Miha Grčar. 2013. URL tree: efficient unsupervised content extraction from streams of web documents. *CIKM, ACM.*

[6] Matthew E. Peters and Dan Lecocq. 2013. Content extraction using diverse feature sets. *WWW, ACM* .

[7] Guo, Y., Tang, H., Song, L., Wang, Y. and Ding, G. 201*0.* 'ECON: An approach to extract content from Web News Page'. *APWEB, IEEE.*

[8] Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. 2010. Boilerplate detection using shallow text features. *WSDM, ACM.*

[9] Dandan Song, Fei Sun, Lejian Liao, 2013. A hybrid approach for content extraction with text density and visual importance of DOM nodes. *KAIS Journal, SPRINGER.*

[10] Zhan Bhu, Chenguchi Zhang, Zengyou Xia, Jiangdong Wang. 2013. An FAR-SW based approach for webpage information extraction. *Inf. Syst. Front. Journal, SPRINGER.*