Bauhaus-Universität Weimar
Faculty of Media
Degree Programme Computer Science and Media

Building a Corpus for Hyperpartisan News Detection

Master's Thesis

Payam Adineh

∞↙ Referee: Prof. Dr. Benno Stein

Submission date: September 12, 2018

Declaration

Unless otherwise indicated in the text or references, this thesis is entirely the product of my own scholarly work.

Weimar, September 12, 2018

. . . .

Payam Adineh

Abstract

As the polarization of societies increases, news publishers that cater to specific extreme views become attractive to more people. Such hyperpartisan news publishers, however, tend to draw a simple picture of their views being the only truth, thus furthering the divide in the society. As a small step towards countering this trend, this thesis aims to provide the tools to distinguish hyperpartisan news from impartial ones. These tools will enable others to investigate, monitor, and act upon the spread and effects of hyperpartisan news.

The main contributions of this work are the construction of a corpus of 1.5 million political news articles from 358 different news publishers and a detailed analysis of this corpus with regards to the use of hyperlinks within and between the publishers, the reuse of articles, and an automatic classification of the articles by publisher bias.

Contents

1 Introduction 1

2 Related Work 4

2.1 Fact Checking and Misinformation 4

2.2 Ecosystem of Partisan Websites 4

2.3 Hyperpartisan and Fake News Detection 5

3 Article Collection 6

3.1 News Producers Discovery 6

3.1.1 BuzzFeed Partisan News Sites List 6

3.1.2 Media Bias/Fact Check (MBFC) 7

3.1.3 Buzzfeed And MBFC Overlap 8

3.1.4 NewsIR16 Corpus 9

3.2 Article URL Collection 10

3.2.1 Facebook Pages 10

3.2.2 Sitemap 11

3.2.3 Link Statistics 13

4 Corpus Construction 19

4.1 Archiving 19

4.1.1 Tool 19

4.1.2 Checking The Producers 21

4.1.3 Archive Quality Assurance 23

4.2 Distributed Storage 28

4.2.1 Hadoop Distributed File System 29

4.2.2 MapFile 29

4.2.3 MapFile Creator 29

4.3 Distributed Archiving 30

4.3.1 Task Control Automation 30

4.4 Main Content Extraction 32

4.4.1 Writing the Wrappers 32

4.5 Corpus Formatting . . . 34
4.5.1 JSON Line . . . 35
4.5.2 XML . . . 35

5 Analysis 37
5.1 Corpus Statistics . . . 37
5.2 Corpus Anchor Element Analysis . . . 39
5.3 Duplicate Article Detection . . . 39
5.3.1 Text Reuse Analysis Pipeline . . . 41
5.4 Classification Experiment . . . 42
5.4.1 Logistic Regression . . . 42
5.4.2 Naive Bayes . . . 46
5.4.3 Random Forest Classifier . . . 47
5.4.4 Experiment on SemEval 2019 Corpus . . . 49

6 Conclusion 50

Bibliography 52

The conscious and intelligent manipulation of the organized habits and opinions of the masses is an important element in democratic society. Those who manipulate this unseen mechanism of society constitute an invisible government which is the true ruling power of our country[2005].

– Edward Bernays

Acknowledgements

My sincere thanks go to Johannes Kiesel and Dr. Martin Potthast, who guided me through the whole work.

I also want to thank my family for the continuous love and encouragement, and my colleagues for their valuable support Sarah Alburakeh, Masoud Allahyari, Milad Alshomary, Alexander Bondarenko, Negin Yaghoobisharif.

Thanks to everyone who made a mark in my life!

Chapter 1

Introduction

Partisanship, in the world of journalism and news producers, refers to the ideologically biased creation and distribution of news by authors and organizations. To be more precise, partisan journalists present their perspective on political incidents in a way that it seems they are living in a parallel universe, in which every step and action that their favorable party takes is absolutely and unmistakably right. Hyperpartisanship has similar meaning, and is used especially when news producers extremely manipulate the coverage of the reality with a tendency to the right or left wing political parties

These days, partisanship, in addition to the widely spread fake news, has resulted in a huge threat to the political awareness of people, by publishing not only partially manipulated versions of reality but also totally made-up and untruthful stories, as well. According to Mitchel Stephens, a journalism professor at New York University, this is not new: "Journalism in the United States was born partisan and remained, for much of its history, loud, boisterous and combative"[1]. With the advent of the Internet and social media, these hyperpartisan news producers have become more powerful in the matter of distribution and consequently more influential on public opinion. Partisanship and fake news have become highly prevalent, widely known and controversial, specifically after the 2016 United States election. There has been much research on the effects of such news on the aforementioned election, and how it led the society to a massive political division. However, this partisan separation is of great importance for the societies; thinking thoroughly about it, one can realize a continuous spectrum of different thoughts is an ultimate demand to enjoy a decent democracy and without that people can only hear one voice of totalitarianism. Consequently, partisanship has its merits for society; Nev-

[1] https://www.politico.com/magazine/story/2017/06/26/goodbye-nonpartisan-journalism-and-good-riddance-215305

ertheless, people ultimately need to be aware of this paradox: democracy does not work properly without partisanship, but in the situation that partisanship becomes a top priority and more important than anything else, it can not be considered as an advantage anymore. As political scientist Lilliana Mason [2016] convincingly argues, “The more partisan we become, the more emotionally we react to normal political events.” And when emotions are heightened, everything becomes a threat to status. Politics becomes more about anger. And, here’s the warning from Mason: “The angrier the electorate, the less capable we are of finding common ground on policies, or even of treating our opponents like human beings.”[2]

Moreover, biased journalism affects people’s life from both the political and nonpolitical side; They not only manipulate people’s minds, but also affect their identity. In 2009, Sean Westwood, conducted research over National Election Study, a survey that collects Americans’ political opinions and behavior[]. “I didn’t expect the political conflict to spill over from political aspects of our lives to nonpolitical aspects of our lives, and I saw that happening in my social group, Partisanship, for a long period of time, wasn’t viewed as part of who we are,” he also mentioned. “It wasn’t core to our identity. It was just an ancillary trait. But in the modern era, we view party identity as something akin to gender, ethnicity or race - the core traits that we use to describe ourselves to others.”. Considering this vast impact of extreme partisanship on the society, it is necessary to find a solution to both decline the consequences and increase the awareness.

In this thesis, we focus on creating a platform to recognize and distinguish extremely biased news and their orientation from unbiased ones. However, to follow that direction one needs to have a proper dataset of news articles, which has not existed thus far; therefore, we create the required dataset. To do so, we chose more than 600 news producers whose orientation was labeled in 5 different categories, from the left to the right bias, by BuzzFeed[3] and Media Bias Fact Check[4]. The next step was to extract news articles links from those news producers, and afterwards all the links were crawled to create a large-scale corpus of approximately 1,5 million articles for the further study and analysis. It is worth mentioning that in the field of Computer Science, there have been several studies in this regard. For comparison, in Potthast et al. 2017 from Webis group of Bauhaus-Universität Weimar, such research

[2] https://www.vox.com/the-big-idea/2017/9/5/16227700/hyperpartisanship-identity-american-democracy-problems-solutions-doom-loop

[3] https://www.buzzfeed.com/craigsilverman/inside-the-partisan-fight-for-your-news-feed

[4] https://mediabiasfactcheck.com/

was done on a corpus of 1,627 articles from 9 different publishers.

This thesis aims to achieve two major objectives. Firstly, creating a corpus of news articles from the variety of news producers that produce political news with their specific point of view in the United States. Secondly, analyzing the corpus with different learning methods in order to train a model which is able to detect biased news articles with a high accuracy.

1. The process of collecting articles and constructing the corpus:
    - Find political partisan news producer.
    - Find political news articles using the producers' Facebook feed and sitemap
    - Crawl and archive all the news articles while continuously monitoring the archive quality.
    - Extract the main information from the archived articles, such as title, author, time, content.
2. Analysis the corpus:
    - Analyze the main information to recognize duplicate and near duplicate documents.
    - Train classifiers with different machine learning models to distinguish extreme bias and also articles orientation.

Chapter 2

Related Work

In this chapter, we describe an overview of several studies that we found related to our topic. These related works will be introduced in different sections. First, we present researches about fact-checking and misinformation detection, then studies about hyperpartisan and fake news detection.

2.1 Fact Checking and Misinformation

In the contemporary world of journalism, a rapid growth of misinformation is considered as a huge concern, which requires attempts to enhance the way that we detect and respond to misinformation. A research by Zhang et al.[2018] was done regarding this subject. They used 40 of the most shared articles on social media for this research. Moreover, they employed 6 skilled annotators, 3 for content and 3 for context annotation. Then, they offered a set of initial indicators for article credibility in two major categories content and context signals. Content signals can be found from the text of an article, and a context signal can be found from external sources or metadata of an article. In the next phase, to analyze the annotation data, they focused on two measures: "(1) How much annotators agreed with one another when identifying indicators, and (2) how much the annotators' assessments of overall article credibility agreed with domain experts' assessments." Finally, they offered a template for creating a standardized set of indicators for evaluating content credibility.

2.2 Ecosystem of Partisan Websites

The 2016 US Presidential election led to a confusion about the factors that affected Trump's victory. There are several aspects such as socio-economic, cultural, political and technological to discuss. One of the most important answers to that confusion is the creation of a fabricated news sites ecosystem. BuzzFeed News's Craig Silverman did a study[1] about this discussion, which inspected 667 hyperpartisan websites and their corresponding Facebook pages in the last three months before US presidential election. They realized 20 top-performing false stories about the election from hyperpartisan sites generated 8,711,000 shares, reactions, and comments on Facebook. On the other hand, 19 major news websites generated 7,367,000 in the same period of time. In addition, they mentioned that the top election content from major outlets had been outperformed by fake election news on Facebook. This study also indicates that hundreds of these websites are begin run by now-famous Macedonian teens[2], and in one example a Facebook page which is run by a Macedonian teenager frequently outperforms some of the larger conservative pages operated by Americans. In addition, Bhatt et al.[2018] in another similar research reached to the conclusion that lots of these news websites that were being established during the election campaign time were then abandoned after the election. They also indicate that this ecosystem directs users traffic by creating a link from one site to another and liking each other's pages and posts.

2.3 Hyperpartisan and Fake News Detection

In this regard, Potthast et al. [2017] is an good example, which employed the BuzzFeed-Webis fake news corpus with 1,627 articles from 9 different news producers. These articles have been fact-checked by BuzzFeed journalists. They also are categorized into three categories namely, main-stream, left-wing and right-wing. In this approach, they used common features in a combination with ones related to the news domain. The part-of-speech and n-grams are the common features, and the ratios of quoted words and external links, the number of paragraphs and their average length are the domain specific features. They utilize several baseline models including, a top-based bag-of-words model, a model using only the domain-specific features and finally naive baselines. They concluded that with style analysis hyperpartisan news from unbiased news (F_1=0.78), and satire from both (F_1=0.81) are distinguishable. It is also mentioned that left-wing and right-wing news enjoys more stylistic similarity compared with ones from the mainstream.

[1] https://www.buzzfeednews.com/article/craigsilverman/viral-fake-election-news-outperformed-real-news-on-facebook

[2] https://www.buzzfeednews.com/article/craigsilverman/how-macedonia-became-a-global-hub-for-pro-trump-misinfo

Chapter 3

Article Collection

The first step to creating a corpus of partisan news articles is to find a straightforward method to collect the articles. There are lots of obstacles in this regard, for instance, how do we find the news producers? how do we discover URLs of news articles? how do we make sure to only collect the political articles? In this chapter, we describe our solutions to overcome these difficulties.

3.1 News Producers Discovery

Finding general information about political news producers was our first priority. We obliged to depend on some reliable sources of information to maintain a list of news producers in addition to their political view preferences. Media Bias Fact Check and BuzzFeed are two sources that were used for this purpose.

3.1.1 BuzzFeed Partisan News Sites List

BuzzFeed is a cross-platform media network, which assumes that they have both infrastructures of a tech company and innovation of being a culturally obsessed corporation. BuzzFeed concentrates on shaping a media platform for today's world, and the future. For this thesis, an article from BuzzFeed is used which is described in Section 2.2. As a part of the study they provide a list[1] of political news producers, plus extra information including the bias category and the Facebook page of each producer-if they have any-. This list contains 677 political news websites with information as exemplified in Table 3.1.

In this list, bias categories are shortened into the two major groups of conservative and liberal. Moreover, we need to find the articles on the websites,

[1] https://github.com/BuzzFeedNews/2017-08-partisan-sites-and-facebook-pages/blob/master/data/all-partisan-sites.csv

Site	Political Category	Facebook Id
100percentfedup.com	right	311190048935167
21stcenturywire.com	left	182032255155419
24dailynew.com	right	515629708825640
aattp.org	left	108038612554992

Table 3.1: Excerpt list of BuzzFeed news producers

but every website looks different. To find the articles in these producers, checking the Facebook pages and going through the sitemaps are two different approaches that were applied. First, all of the items in the list are checked for Facebook Id entry. Following that, all the pages which have an active Facebook page were labeled and also all the other items that don't have a Facebook page are examined for the availability of sitemap. As Table 3.2 shows that out of 490 producers that are supposed to possess a Facebook page, 351 items own an active and available Facebook page and the other Facebookpages are removed or deactivated. In addition, 107 news producers websites have sitemaps. The other 219 entries are not used for the next stage of the thesis owing to either unavailability of their website or duplication of producers in the BuzzFeed list. Table 3.2 displays that 48 items on the list are duplicate entries and 171 websites are not available anymore.

Sites	Left	Right	Total
Total Number	178	499	677
Duplicate	9	39	48
Unavailable	31	140	171
Facebook	120	231	351
Sitemap	18	89	107

Table 3.2: Availability of the Facebook pages and sitemap of the BuzzFeed list

3.1.2 Media Bias/Fact Check (MBFC)

MBFC is a fact-checking website which labels websites according to their political view in different categories from right to left wing bias. As mentioned on the website their objective is to inspire action and a rejection of overtly biased media and moreover to return to an era of straightforward news reporting. However, their index includes a huge list of websites from all over the world that focused on a variety of topics like politics, economy, and sport to name but a few. MBFC categorized websites into five major categories, namely extreme left, left-center, least biased, right-center, and extreme right. Since we had enough websites for extreme left and right from the BuzzFeed list, we only focused on both right-center and left-center, in addition to the least bias websites. However, due to the lack of left-wing articles, we decided to find new left sources from MBFC. In this website for each category, there is a large list of websites, and for each of them, they provide a detail page which has brief information about the producer. Therefore, we needed to go through the detail page to check if it fits our criteria.

Criteria:

1. It must cover the United States news.
2. It must have a political section.
3. The website must own a proper sitemap.

We found that 109 websites out of 1310 fulfill our criteria and are thus included in our corpus. Table 3.3 illustrates the statistics of websites that were checked during this phase of the thesis.

	Left	Leftcenter	Least	Rightcenter	Total
All Subjects	290	447	362	211	1310
Political	12	49	28	20	109

Table 3.3: Media Bias/Fact Check indexed sites statistics

3.1.3 Buzzfeed And MBFC Overlap

During the process of finding news producers from MBFC, we also check for the producers that both Buzzfeed and MBFC share in common, however, these two have a disagreement over labeling some of the producers. Considering this ambiguity, we decided to collect the articles but not to include them in the corpus. One can find the list of these news producers which have different labels in BuzzFeed and MBFC in the Table 3.4.

News Producer	MBFC	BuzzFeed
consortiumnews.com	least	left
amgreatness.com	rightcenter	right
rare.us	rightcenter	right
theamericanconservative.com	rightcenter	right
washingtonexaminer.com	rightcenter	right
liberalmountain.com	leftcenter	left
billmoyers.com	leftcenter	left
mintpressnews.com	leftcenter	left
secondnexus.com	leftcenter	left

Table 3.4: MFBC and BuzzFeed disagreement over bias label

3.1.4 NewsIR16 Corpus

The Signal Media One-Million News Articles Dataset[2] was released by Signal Media to assist research on news retrieval. The corpus is in JSON format and each article has the following fields:

- id: a unique identifier for the article
- title: the title of the article
- content: the textual content of the article (may occasionally contain HTML and JavaScript content)
- source: the name of the article source (e.g. Reuters)
- published: the publication date of the article
- media-type: either "News" or "Blog"

For several apparent reasons, we decided not to use this corpus. Firstly, it only covers news from September 2015 that make this corpus too specific. Secondly, it contains news articles from a huge variety of publishers around the world, and Thirdly, it contains not only political news, which does not serve our task to collect the articles relevant to US politics.

[2] `http://research.signalmedia.co/newsir16/signal-dataset.html`

3.2 Article URL Collection

The next stage after discovering the news producers is to find URLs of the news articles. One option to find the URLs would be to use a crawler which starts from the homepage of the news websites and extracts all the internal links that it finds, then the crawler adds them to a queue and continues crawling all the pages and add new links to the queue. However, using a crawler is time-consuming and has its own technical difficulties. Therefore, we decided to go in another direction and to use a shortcut, we used the Facebook page and sitemap of the news producers. We assume that news producers which own a Facebook page, also share their news articles in that medium to influence more people due to the engagement of social media in modern human life. Moreover, most well-design and well-organized websites follow a special standard of using sitemap which contains all the links that one can find in a website.

3.2.1 Facebook Pages

Facebook is a social networking website which was created in 2004 and since then has gained tremendous growth and popularity over time. Due to the popularity of this platform, most of the news producers tried to absorb a new audience as well as keep the current ones by using this medium. As a result, a news producer not only publishes the news on their website, but also they share a link to their new articles on Facebook as well.

Facebook Graph API

By using the so-called Graph API[3] one can access user-related data programmatically. To collect data from Facebook pages we focused on three entities namely, post, comments, and reactions. Basically, we went through the list of Facebook pages that we find in the BuzzFeed list, then we retrieve all the posts that they published. The next step was to collect reactions and comments from the users on each post, which we thought would be useful for other types of research and studies. The information that we collected for each entity can be found in the table 3.5.

We collected the data from the Graph API and stored post, comments, and reactions in a relational database. However, since the process of retrieving comments and reactions was so time-consuming, we decided to deactivate that feature for this thesis.

[3] `https://developers.facebook.com/docs/graph-api/`

Post
Post ID
Content
Link
Creation Time
Page ID

Comment
Comment ID
Content
Creation Time
User ID
Username
Post ID

Reaction
Reaction ID
Username
Type
Post ID

Table 3.5: Facebook entities accessible through the Graph API

3.2.2 Sitemap

A sitemap is a systematic and hierarchical view of the website in an XML format. The sitemap protocol was introduced by Google[4] to standardize and uniform existing approaches. This protocol is quite helpful and practical specifically when it comes to those links that needed a user interaction on a form to generate the links. Considering all the benefits of sitemap protocol, we used this method to find articles' links for all the 109 news producers that we found in the Media Bias/Fact Check. We also used a sitemap for 107 news producers from the BuzzFeed list to compensate for a prevalence of right-wing articles.

Sitemap Link Extractor

In order to develop a tool to extract the links from a sitemap, one needs to be aware of the standard structure of this protocol.

In principle, there are two main types of XML sitemaps which are URL sitemap and Index sitemap.

- The URL sitemap contains the URLs of the website. These files usually are in XML format and in some cases for archiving purposes, publishers stored them in a compressed format of xml.gz.
- The Index sitemap contains a list of all the URL sitemap of that website.

By going through the URL sitemap structure, three more categories of URL sitemap can be found.

- sitemap for web pages which has all the information regarding the links in a webpage.

[4] `https://www.google.com/sitemaps/protocol.html`

- Image sitemap which contains images detail and their URL in a website.
- Video sitemap that includes information about the videos and their corresponding URL.

By considering all the aforementioned information in mind, the tools that we needed to use for extracting the URLs should support all of these details. Moreover, for all of the news producers that we have, we need to collect a lot of extra information as follows:

1. URL.
2. Is it index sitemap?
3. Is it compressed?
4. Does the sitemap show the category of the news articles such that one can directly use only articles from the political section?

The first three items in the lists are obligatory information that is needed to have access to a sitemap. In addition, the last parameter which is not mandatory gives us an ability to filter URLs, due to the factor that we only want political articles and some website also have other subjects to report. For filtering the URLs there are several methods. First, we can use a specific keyword as a filter in a sitemap index, and it will only consider sitemap URLs that contain that keyword which can be seen in figure 3.1. Second, a special keyword can be used in order to restrict the URLs in a URL sitemap and Figure 3.2 is an example. Finally, the last parameter that can be used in order to limit the URLs is using a pair of tag and value which for instance can be seen in figure 3.3.

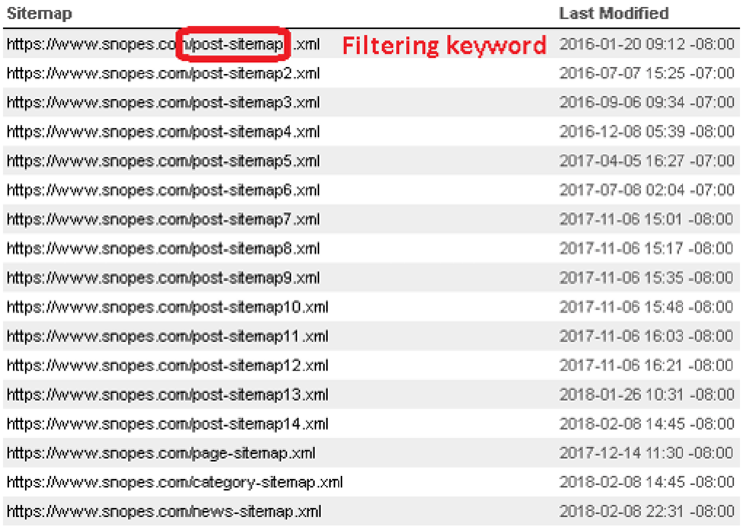
Our implementation of extracting URLs with features that were described can be simply used by running the following command.

```
$ python extract_urls.py --url "http://site.com/sitemap.xml"
--domain "site.com" --siteid "siteid" [--not_index] [--gzip]
[--sitemap_filter "keyword"] [--url_filter "keyword"]
[--tag_value_filter "tag*value"]
```

Command Parameters:

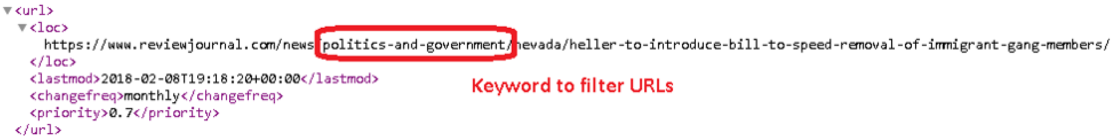
- –url: URL of the sitemap
- –domain: Host of the website

Figure 3.1: A sample for finding a keyword to filter URLs in a index sitemap



Sitemap	Last Modified
https://www.snopes.com/post-sitemap.xml	2016-01-20 09:12 -08:00
https://www.snopes.com/post-sitemap2.xml	2016-07-07 15:25 -07:00
https://www.snopes.com/post-sitemap3.xml	2016-09-06 09:34 -07:00
https://www.snopes.com/post-sitemap4.xml	2016-12-08 05:39 -08:00
https://www.snopes.com/post-sitemap5.xml	2017-04-05 16:27 -07:00
https://www.snopes.com/post-sitemap6.xml	2017-07-08 02:04 -07:00
https://www.snopes.com/post-sitemap7.xml	2017-11-06 15:01 -08:00
https://www.snopes.com/post-sitemap8.xml	2017-11-06 15:17 -08:00
https://www.snopes.com/post-sitemap9.xml	2017-11-06 15:35 -08:00
https://www.snopes.com/post-sitemap10.xml	2017-11-06 15:48 -08:00
https://www.snopes.com/post-sitemap11.xml	2017-11-06 16:03 -08:00
https://www.snopes.com/post-sitemap12.xml	2017-11-06 16:21 -08:00
https://www.snopes.com/post-sitemap13.xml	2018-01-26 10:31 -08:00
https://www.snopes.com/post-sitemap14.xml	2018-02-08 14:45 -08:00
https://www.snopes.com/page-sitemap.xml	2017-12-14 11:30 -08:00
https://www.snopes.com/category-sitemap.xml	2018-02-08 14:45 -08:00
https://www.snopes.com/news-sitemap.xml	2018-02-08 22:31 -08:00

Figure 3.2: A sample for finding a keyword to filter URLs in a URL sitemap



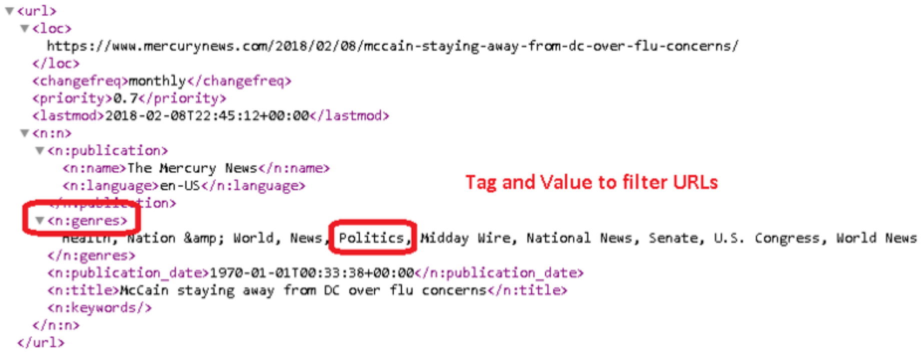
```
▼<url>
  ▼<loc>
     https://www.reviewjournal.com/news/politics-and-government/nevada/heller-to-introduce-bill-to-speed-removal-of-immigrant-gang-members/
   </loc>
   <lastmod>2018-02-08T19:18:20+00:00</lastmod>
   <changefreq>monthly</changefreq>
   <priority>0.7</priority>
 </url>
```

- –siteid: Identification number for the website
- –not_index: This parameter is optional and is needed when URL parameters is not referring to the index sitemap
- –gzip: This parameter should be used only in the situation that sitemap URL is in compressed format
- –sitemap_filter: This parameter can be used, if a user wants to apply filter on the list of URL sitemap in an index sitemap
- –url_filter: This parameter is useful for filtering URLs in a URL sitemap
- –tag_value_filter: To apply filter on the metadata in a URL sitemap using a key and value, users can use this parameter

3.2.3 Link Statistics

In this section, we focus on a basic analysis of data that we collect using our two major techniques to find articles' links from news producers. Considering the

Figure 3.3: A sample for choosing a tag and value to filter data in a URL sitemap



```
▼<url>
  ▼<loc>
     https://www.mercurynews.com/2018/02/08/mccain-staying-away-from-dc-over-flu-concerns/
   </loc>
   <changefreq>monthly</changefreq>
   <priority>0.7</priority>
   <lastmod>2018-02-08T22:45:12+00:00</lastmod>
  ▼<n:n>
    ▼<n:publication>
       <n:name>The Mercury News</n:name>
       <n:language>en-US</n:language>
     </n:publication>
    ▼<n:genres>
       Health, Nation &amp; World, News, Politics, Midday Wire, National News, Senate, U.S. Congress, World News
     </n:genres>
     <n:publication_date>1970-01-01T00:33:38+00:00</n:publication_date>
     <n:title>McCain staying away from DC over flu concerns</n:title>
     <n:keywords/>
   </n:n>
 </url>
```

fact that for BuzzFeed producers we apply both techniques and for MBFC we only used the sitemap approach, these statistics will be introduced separately in two subsections.

BuzzFeed producers

In the process of link extraction from the news producers that we found on BuzzFeed websites, 458 news producers were involved. We managed to extract links from the Facebook pages for 351 news producers. Furthermore, for the other 107 producers, we used sitemaps to discover the links.

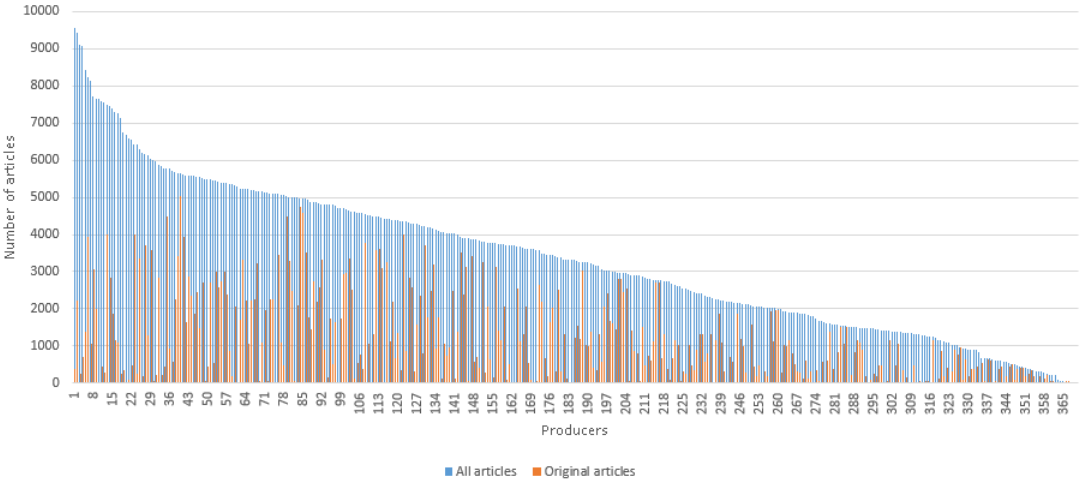
Facebook Pages

By using Facebook Graph API, we were able to find 1,239,955 posts for both liberal and conservative news producers. During this process, we found out that not all the links belong to those news producers, since in lots of cases they have shared contents from other sources which we called them third-party sources in this thesis. This incident happened especially when the content suits their political perspective, and as you can see in table 3.6, approximately two-thirds of the posts are not produced originally by the actual publishers. However, this proportion is slightly different for left and right wing parties. the actual publisher produced 37 percent of the content shared on Facebook pages of left-wing pages; moreover, 67 percent of news shared by right-wing websites are shared from third-party news producers. Finally, we only considered the original articles from news producers, which amounts to 423,764 news articles; the share of the left party is 169,091 articles, and 254,673 news articles are the original ones for conservative producers. In figure 3.4, the proportion of the All news articles to Original news articles can be found, in which the blue color represents all the posts including third-party sources and orange color is used for original article; It also shows all the producers that we have investigated with Facebook graph API approach.

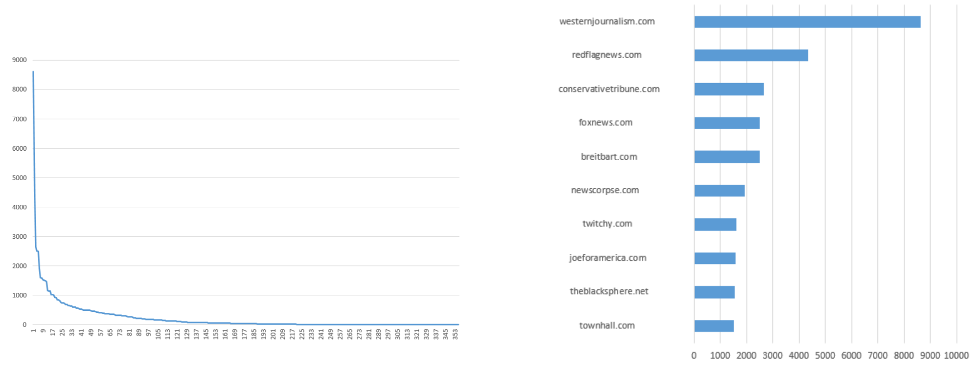
	Absolute			Average per site		
Number of articles	Left	Right	Total	Left	Right	Total
Third-Party	288,552	527,639	816,191	2,327	2,162	2,217
Original	169,091	254,673	423,764	1,409	1,102	1,207
Total	457,643	782,312	1,239,955	3,690	3,206	3,369

Table 3.6: Article statistics from the BuzzFeed list using Facebook Graph API

Figure 3.4: Number of all news article to the original ones per site



We also went a bit further in this analysis, as we wanted to be well-informed about the following facts: Which third-party sources were referenced more by our producers? How often did our producers reference one another? For instance, figure 3.5a indicates how often news producers have shared content from the other producers in our list, and figure 3.5b shows the list of top 10 news producers that produced the most viral news articles. As one can see, westernjournalism.com with a huge margin is at the top of the list and it seems that they produced more content that is worth sharing by the others. The other significant detail about this list is that 90 percent of it consists of conservative websites and the only liberal producer is newscorpse.com. From this statistics, we can conclude that conservative websites have a more organized network to share one another materials.



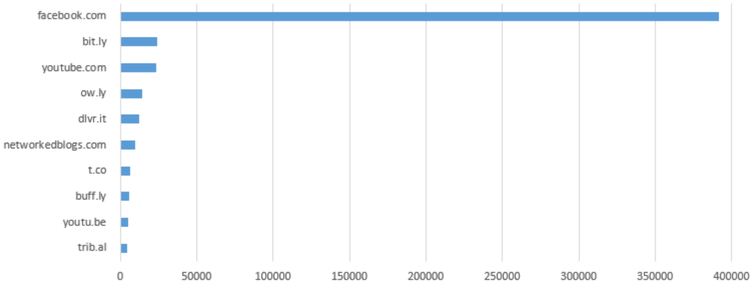
(a) BuzzFeed producers shared by other BuzzFeed producers

(b) Top 10 BuzzFeed list news producers referenced by the other BuzzFeed producers

Figure 3.5: Statistics of publishers being referenced by other news producers in the BuzzFeed list

Figure 3.6 displays the top 10 third-party sources that their content was shared by the producers on our list. Moreover, as is obvious from the list, there is no partisan news producer on the list that we missed to include in this study. In addition, it also shows, links to the Facebook pages and Youtube[5] videos, and also services to shorten URLs are very common to use in the Facebook's posts.

Figure 3.6: Top 10 third-party sources that produced content which is shared by the BuzzFeed news producers list



Sitemaps

To continue the links extraction process, we applied the sitemap approach to discover links to news articles for 107 BuzzFeed news producers; we only

[5] http://youtube.com

tried to apply this approach on the news producers that didn't have a Facebook page or we could not find any article for them using Facebook Graph API. Table 3.7 shows, 972,866 articles were found during this process. However, some extra filtering over the obtained data was required, since there were a great number of links, including tags, authors homepages, and categories, which do not point to a news article. Finally, we found 579,208 links, 70,279 links from the left-wing party and the other 508,929 from the right-wing party. On average for each site, we discovered 5,413 articles. Moreover, if we go into more detail, the average number of liberal articles per site is 3,904 and conservative producers have 5,718 articles on average.

	Absolute			Average per site		
Number of articles	Left	Right	Total	Left	Right	Total
Original	70,279	508,929	579,208	3,904	5,718	5,413
Total	101,273	871,593	972,866	5,626	9,793	9,092

Table 3.7: Article statistics of the BuzzFeed list using sitemap approach

Media Bias/Fact Check producers

In addition to the articles obtained through the BuzzFeed list, we obtained 3,191,915 articles from 109 news producers of Media Bias/Fact Check. As Table 3.8 illustrates Left-center biased producers have 1,392,187 articles and on average each producer possesses 28,411 news articles. The share of least bias and right-center producers was 750,872 and 420,218 news articles. Least bias websites have on average 26,816 pieces of news, and right-center ones have 29,283 articles. Finally, out of 12 news producers that we found in the MBFC owing to the reason that we didn't have enough articles for the left wing party, we discovered 628,638 links in general which become 52,386 article on average for each producer.

Final Statistics

After article collection the corpus contains approximately 4.2 million URLs. Table 3.9 shows the number of URLs that are attempted to be archived in Chapter 4.

	Absolute				
Number of articles	left	Left-center	Least	Right-center	Total
Total	628,638	1,392,187	750,872	420,218	3,191,915
	Average per site				
Number of articles	left	Left-center	Least	Right-center	Total
Average	52,386	28,411	26,816	29,283	26,425

Table 3.8: Article statistics found from MBFC news producers

	Absolute					
Number of articles	left	Leftcenter	Least	Rightcenter	Right	Total
BuzzFeed Facebook	169,091	-	-	-	254,673	423,764
BuzzFeed Sitemap	70,279	-	-	-	508,929	579,208
MBFC Sitemap	628,638	1,392,187	750,872	420,218	-	3,191,915
Total	868,008	1,392,187	750,872	420,218	763,602	4,194,887

Table 3.9: Corpus statistics after article collection

Chapter 4

Corpus Construction

In this chapter, the process of creating the corpus is described in detail. It includes several steps to deal with different obstacles that we encountered, and also our solution to solve each problem. After the article collection phase, we had 4,194,887 news articles to crawl for the further use in this study. Accordingly, we need a reliable method to both crawl and store the articles. Moreover, we need to find the main content in the articles and store all the important fields of the articles in the formats that we want to construct the corpus.

4.1 Archiving

We want to store all of the web pages that we found during article collection in our storage owing to the reason that a lot of web pages are disappearing from the Internet. Accordingly, we want to archive them in a way that we can access the web page as we have access to them when they are still available and online. For this purpose, the most significant concern for us is the quality of the archived page and having access to every part of a web page from the text to the media that producers used in their articles.

4.1.1 Tool

There are several ways to crawl a web page from the Internet and which one to choose absolutely depends on the kind of content that one wants to archive. Here we briefly describe the four major approaches[1] to archive a web page, and moreover the reason that we have chosen one to archive our web pages.

[1] `https://www.labnol.org/internet/archive-web-pages/20192/`

Archiving Methods

Text Content
There are many application and browser extensions like Evernote[2] and OneNote[3] which help users to save the text of a webpage and make it available to access the content through the application.

Packaged
Packaged archiving is a simple way to have access to the content of a web page without a demand for an additional extension. Storing web pages in a PDF format is one example and PDF writers exist in the features of modern browsers.

Local Copy
All the modern browsers have a feature to store a web page entirely in the local machine. Also tools like wget is a good example to save a URL, however, it stores nothing more than the HTML content.

Web ARChive
Internet archives like `http://Archive.org/web/` and `http://Archive.is/` are websites that store an archived version of a URL including all the Javascript files, and all the other assets in a Warc format. It is possible to download a Warc file from Archive.is, but the problem is that it doesn't have all the pages that users are looking for.

Choosing the Method
Considering the brief description that is mentioned above, one can see the latest method provides us with the highest quality of archiving a page, due to the fact that we can store an entire page and access content in the way that it originally existed on the Internet. In addition, we already have a well-implemented archiving tool in the Webis group[2018].

Webis Web Archiver
Using simple methods like just crawling the HTML content of a page is not enough to have exact results as we normally see in the actual page. This happens for several reasons, for instance, web pages have so many external resources like CSS files or JavaScript files and also multimedia files which do not exist in the HTML page. In addition, in some cases, the web page needs

[2] `https://evernote.com/`

[3] `https://www.onenote.com/`

to load the content by calling client-side scripts. As a result, we needed to find a way to solve all of these issues. Therefore, we employed Webis Web Archiver[2018] as a required tool which simulates modern browsers' behavior to create a standard WARC web archive. The WARC files which are produced by Webis Web Archiver are compatible with the other third-party tools. This tool has also introduced another vital feature to render and reproduce a web page using its WARC file. Moreover, archiving and reproduction can be done as follows.

```
$./archive.sh --url "url" --scriptsdirectory "directory" --script
"script" --output "output"
$./reproduce.sh --url "url" --archive "directory" --scriptsdirectory
"directory" --script "script" --output "output"
```

Parameter Definition:

- **–url:** The URL of the web page that the user wants to archive or reproduce.
- **–scriptsdirectory:** The path to the directory where the user stores user simulation scripts
- **–script:** The name of the script class.
- **–output:** Path to the directory where the user wants to store the result.
- **–archive:** Path to the WARC file that the user want to reproduce.

4.1.2 Checking The Producers

In our list, we have a huge number of news producers and news article to crawl and archive. To begin with, we wanted to make sure our archiving software works properly, also we wanted the output to be satisfying and in the way that we expected. Therefore, from each news producer, we randomly chose 3 samples to crawl. Then, after finishing the crawling process, we investigate the output. Table 4.1 displays different types of issues that we faced during this process.

Problem Definition:

- **Modal Windows:** In 140 of websites, we have extra windows open on the top of the news articles. For instance, modal windows for newsletter application, petition, and advertisement have occurred the most often. Figure 4.1 shows four examples of modal windows.

Problem	Number Of Sites
Modal Windows	140
Down	68
Short URLs	7
Share Others	5
Read More	3
Homepage	3
Captcha	2

Table 4.1: Archiving Errors

- **Down:** After crawling these samples from each news producer, we realized 68 of them are not available anymore. For some instances, the server was out of order and down and also for some others even the domain was for sale.

- **Short URLs:** There are 7 websites in the list in which all the URLs that we collected are in short format.

- **Share Others:** 5 websites in the list only share news from other producers.

- **Read More:** In 3 websites when a user sends a request for a news article, the article does not load completely and the user still needs to click on read more button to access the entire article. Figure 4.2 displays one example of this kind.

- **Homepage:** All the links that we collected from 3 websites are pointed to the homepage of the website.

- **Captcha:** Only in two samples, we encountered a request for a captcha.

Solution:
Out of 220 problematic news producers, we found a solution for two groups of problems namely, modal windows, and read more buttons. For all of these additional windows, there is a close button that should be pressed to remove the windows from the screen. In addition, for read more button it is absolutely clear that users have to click on the button. Consequently, we decided to simulate user interaction during the archiving process. To do so, we load a page, the next step is to wait for the scrolling action to be finished, then we simulate the user click on necessary buttons. For this purpose, we run

javascript codes. All of these 143 producers have their specific layout and they are totally different from one another. Therefore, in this stage, we rechecked all of these websites and write down the necessary javascript codes to do the required action. Finally, in the code according to the requested host, we can simply decide which ones to execute. In the following code, you can see two examples for two different websites `http://americasfreedomfighters.com/` and `http://angrypatriotmovement.com/`, and their required javascript code to close the modal windows.

```
domain = uri.getHost();
switch (domain) {
case "americasfreedomfighters.com":
        code = "document.querySelector('#modaal-close').click();";
        break;
case "angrypatriotmovement.com":
        code = "document.querySelector('#revexitcloseme').click();";
        break;
}
```

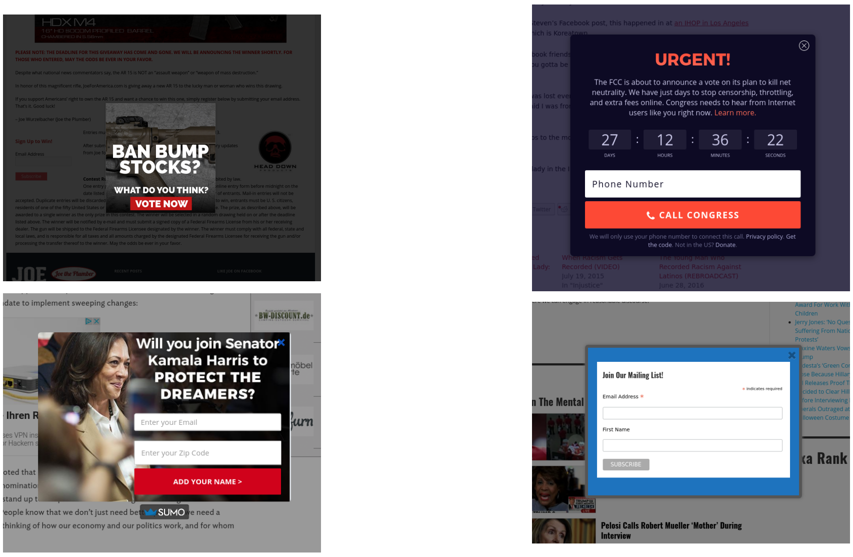
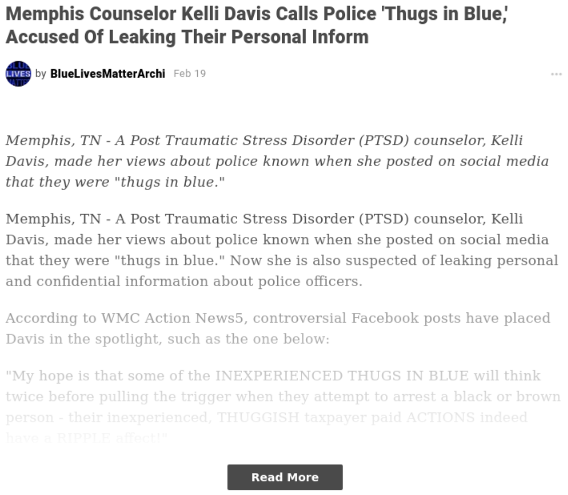

BAN BUMP STOCKS?
WHAT DO YOU THINK?
VOTE NOW
URGENT!
The FCC is about to announce a vote on its plan to kill net neutrality. We have just days to stop censorship, throttling, and extra fees online. Congress needs to hear from Internet users like you right now. Learn more.
27 : 12 : 36 : 22
Phone Number
CALL CONGRESS
Will you join Senator Kamala Harris to PROTECT THE DREAMERS?
Enter your Email
Enter your Zip Code
ADD YOUR NAME >
SUMO
Join Our Mailing List!
Email Address *
First Name
SUBSCRIBE
Pelosi Calls Robert Mueller 'Mother' During Interview

Figure 4.1: Modal Windows Examples

4.1.3 Archive Quality Assurance

Archive quality check is one of the important steps of creating a high-quality corpus. Due to the reason that we had a huge number of web pages to crawl,

Figure 4.2: Read More Example



Memphis Counselor Kelli Davis Calls Police 'Thugs in Blue,' Accused Of Leaking Their Personal Inform

by **BlueLivesMatterArchi** Feb 19

Memphis, TN - A Post Traumatic Stress Disorder (PTSD) counselor, Kelli Davis, made her views about police known when she posted on social media that they were "thugs in blue."

Memphis, TN - A Post Traumatic Stress Disorder (PTSD) counselor, Kelli Davis, made her views about police known when she posted on social media that they were "thugs in blue." Now she is also suspected of leaking personal and confidential information about police officers.

According to WMC Action News5, controversial Facebook posts have placed Davis in the spotlight, such as the one below:

"My hope is that some of the INEXPERIENCED THUGS IN BLUE will think twice before pulling the trigger when they attempt to arrest a black or brown person - their inexperienced, THUGGISH taxpayer paid ACTIONS indeed have a RIPPLE affect!"

we extremely felt the need of a feature to make sure that the archiving process had worked absolutely fine. To do this, we employ a feature of archiving tool to perform this action. This method[2018], utilizes machine learning in order to predict the quality by comparison features that are extracted from the archive and reproduction screenshots. To label the data for the classification stage, human annotators evaluate 6,386 of the 10,000 pages in the Webis Web Archive 17. The annotators score the similarity of the archive screenshots and reproduction in the rage of 1 to 5; Score of 1 means part of the page are just moved up or down a bit and 5 means that main content is missing and the page is not usable. In addition, Deep convolution neural networks are the machine learning method that has been applied to train the classifier. To extract the features, the first step is to crop the images to 4098 pixels, then they indicate the image is converted to grayscale. The next step is to scale down the image to 384x128 pixels. Then, for this size, to match the receptive field of the neurons in the constitutional layers of the network, icons with the usual width of 32 pixels are scaled down to a width of 3 pixels. Finally, using the aforementioned model and required code, we are able to predict the similarity between a pair of the archive and reproduction screenshots.

Archive Quality Check Of Producer Samples

To make sure that we can archive articles from all the news producers in a proper manner, we decided to randomly choose 100 samples from each news producer. Then, we archived and reproduced all of them. We performed this operation to realize whether we are able to successfully crawl all the producers and make sure that the data we have collected is reliable. As you can see in table4.2, out of 475 news producers we have 450 producers with at least one successful crawl. In addition, 38,409 news articles out of 44,806 are successfully archived. Moreover, we used the collected data and evaluate them manually and automatically. The other important finding is that 25 websites do not have any successful crawl, out of which 14 of them faced problems during the archiving process, and the other 11 websites are unavailable and their server is down.

Sites / Article	Number
Total Sites	475
Sites With At Least One Successful Crawl	450
Total Number Of Articles	44,806
Number Of Successful Archived Article	38,309

Table 4.2: Archived samples statistics

Automatic

For this purpose, we collected all the screenshots in two folders. For each article, we have one image in the archive folder and another one in reproduction folder with the exact same name. Then we need to run the following command to predict the result.

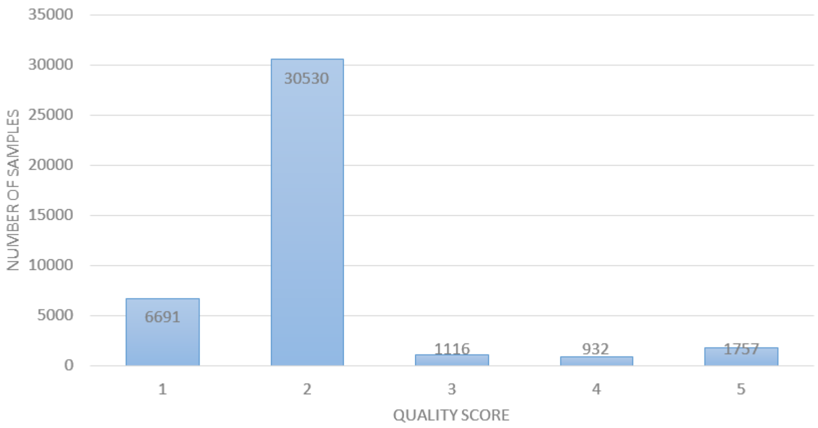
```
$ ./evaluate.sh --imagesa archive --imagesb reproduction --output prediction
```

Parameters Definition:

- **–imagesa:** Path of the folder that contains archived images.
- **–imagesb:** Path of the folder that contains reproduction images.
- **–output:** Path of a file to store the output.

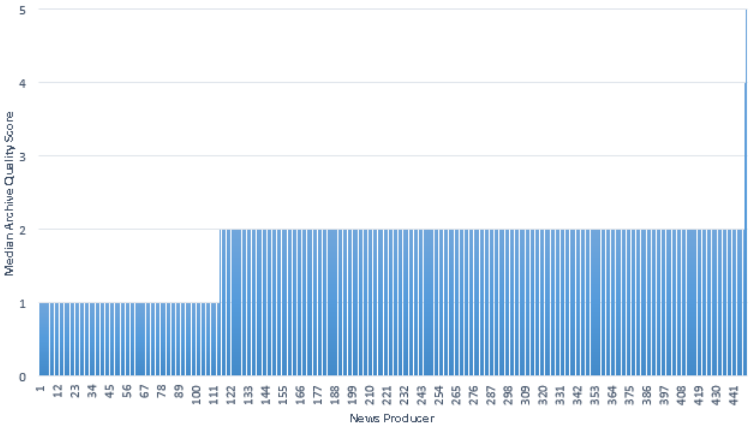
Figure 4.3 displays, almost 96 percent of the samples are categorized with a score of 1 or 2, which means they are mostly similar with some minor differences. A score of 3 means small changes, for example, the comment section of reproduction image is missing and 783 articles are in this category. Category 4 and 5 mean striking difference and unusable page which contains 1,651 articles.

Figure 4.3: Automatic archive quality check scores



In this phase, we also measure the mean archive quality check the score for each news producer. Figure4.4 demonstrates the result of this study. As one can see, only 2 news producers have a mean score of 4 or 5 and 115 number of producers produce identical reproduction image. In addition 333 producers also produce reliable archive and reproduction output.

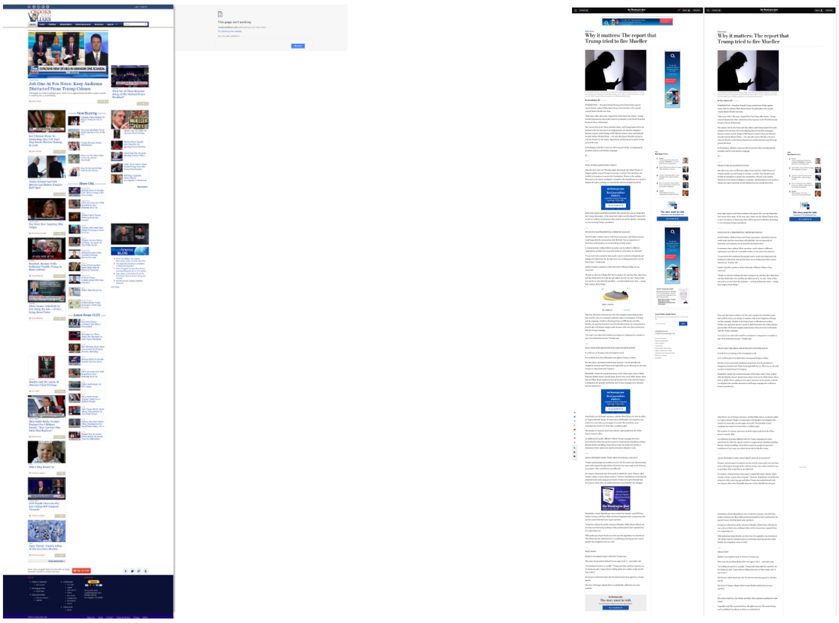
Figure 4.4: Median archive quality score per site



In figures 4.5a and 4.5b two samples of failed and successful archived and reproduced samples are shown.

Manual

To double check the result, we decided to randomly choose 10 instances from the samples that we have already archived and reproduced. We combined



(a) Score 5

(b) Score 1

Figure 4.5: Automatic archive quality samples

each pair of images, side by side in one image. Then we manually checked the images. During this process we realized, there are several cases where the crawled pages are unusable even though the images are the same and archive quality check put them in the first category.

Problematic Sites Statistics

In table 4.3 you can observe, statistics about the number of sites and different error that we found when we inspect archive quality check result. For all the issues that are mentioned in the table we only figure out a way to solve modal windows problem and for the rest of errors, we just ignored the news producers in further phases of the thesis.

Problem Definition:

- **Modal Windows:** In 33 of the websites, we still had the modal windows problem, which their Javascript code was fixed to solve the issue.
- **Park Domain:** For 25 news producers, the result page shows that the domain is for sale. Therefore, the archive quality check works quite well but the crawled page was not what we asked for.

Problem	Number Of Sites
Modal Windows	33
Park Domain	25
Reproduction Problem	8
Gateway Error	3
Archiving Problem	2
Captcha	1
Sitemap Issue	1

Table 4.3: Problematic news producers found by archive quality assurance

- **Reproduction Problem:** For 8 news producers, we were able to archive the page, however, we faced a problem during the reproduction process, which means for these instances archiving does not work properly.
- **Gateway Error:** In output files of 3 producers, we found out it gives us gateway error during the archiving process. Also, in this case, archive quality categorized them in the first category but the crawled page was problematic.
- **Captcha:** Only in one sample, we encountered with a request for captcha. We have the same images in the archive and reproduction process but we cannot use the page.
- **Sitemap Issue:** Finally, one of the news producers has a corrupted sitemap file; all the links in the sitemap pointed to the homepage of the website!

4.2 Distributed Storage

By considering the fact that the process of archiving all web pages is so time-consuming, it was essential for us to store the data in a reliable storage, while we did not want to repeat the process owing to unexpected hardware failures. Therefore, we decided to make use of Webis Betaweb cluster[4].

[4] https://www.uni-weimar.de/en/media/chairs/computer-science-department/webis/facilities/

4.2.1 Hadoop Distributed File System

The Hadoop Distributed File System (HDFS)[2009][5] is a distributed file system which is designed to store very large data sets reliably. HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware.

4.2.2 MapFile

Since we have produced a huge number of files in the archiving process, we did not want to copy all the files directly into HDFS. Therefore, we utilize a format to store files which is more convenient for HDFS for both storage and future computation. MapFile[6] is the format that we used. MapFile is directory which contains two SequenceFile[7] data file and index file. SequenceFile is a persistent data structure for storing pairs of binary key and values, which are append-only and keys are not removable and editable. Data file in a mapfile format have all keys and values records. In addition, index file holds information about each key and starting byte position of the records which adds an extra feature to the data file. This index file works as a lookup file, and due to the small size, it can simply fit in the memory.

4.2.3 MapFile Creator

To create mapfiles, we implement Java program which gets a list of the files as an input and converts all the files to a single mapfile. In this implementation, we put the path of the main directory of each article into the input file. In addition, for each entry, we add 5 files into the mapfile namely, archive.html, archive.png, archive.warc.gz, reproduction.html, and reproduction.png. To generate the keys, we used a combination of article id, URL, and file name, which are separated by space. Our implementation of mapfile creator can be used as follows:

```
$ java -jar ConvertToMapFile.jar InputFile OutPutDirectory
```

Parameter Definition:

- **Inputfile:** List of the root directories of archived articles.
- **OutPutDirectory:** A path to store the mapfile.

[5] https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html
[6] https://wiki.apache.org/hadoop/io
[7] https://wiki.apache.org/hadoop/SequenceFile

4.3 Distributed Archiving

After observing the result of archive quality check, we ignored several problematic news producers; 3,651,229 news articles were left for us to crawl. By using a single machine, we could approximately crawl 600 pages per day; so, with a simple calculation, it would take us about 6,000 days to crawl all pages. To overcome this issue and speed up the crawling, we employed 80 machines from Betaweb cluster. In each machine, we had a list of approximately 45,000 articles to crawl.

4.3.1 Task Control Automation

Due to the reason that manual control of all the 80 machine was so time-consuming and also required a lot of effort, we create a crawling baseline to automatically perform the archiving, quality check, and also mapfile creation.

Archiving and Reproduction

In this part, on each machine, we have a list of URLs to archive. Moreover, we wanted to have three parallel threads on each machine to crawl three different pages simultaneously. To do so, we employ a task manager from Webis Commands. This command uses a text file as a list of task parameter as an input, which has to be named tasks.txt. Also, it requires an executable file next to the tasks.txt file, and this file has to be named task. Then, you should run the following commands as many times as you want to have simultaneous threads.

```
$ webis util task work
```

During the crawling process, we realized there are some cases in which these threads become suspended. To solve this issue, we needed to reset the task manager. Therefore, we used the algorithm 1 to fix this problem. In this algorithm, we check the number of successful and failed tasks every 15 minutes. Then we check if numbers did not change from the last time that we had checked, we reset the task manager.

```
while successfull tasks + failed tasks != total number of tasks do
    Get number of current successful tasks;
    Get number of current failed tasks;
    if current failed tasks = old failed tasks
    AND current successful task = old successful task > then
        Restart the task server;
    else
        old successful tasks = current successful tasks;
        old failed tasks = current failed tasks;
        Wait for 15 minutes;
    end
end
```

Algorithm 1: Algorithm to reset suspended threads

Archive Quality

Archive Quality is employed to check if the archiving process works reliably. To do this, we use algorithm 2 which works on a daily basis. It finds the news archived instances, then it copies all the archive and reproduction images in two separate directories. The next step is to run the archive quality checker over these images.

```
while Archiving task is still running do
    Copy archive and reproduction images in two different folders;
    Run archive quality check;
    Wait for 24 hours;
end
```

Algorithm 2: Daily Archive Quality Check

MapFiles Creation

Creating mapfiles and also copy them to HDFS, was another task that should be handled automatically. To perform this, we find all the successfully archived crawls which are also confirmed by archive quality checker. Then, we create a list of this instances, and every 1,000 items of the list will convert to a mapfile. In the end, this algorithm copies all the generated mapfiles to HDFS.

```
while Archiving task is still running do
    Recognize news successfully finished tasks;
    Create lists of maximum 1000 entries;
    Convert last step lists to mapfiles;
    Copy the mapfiles to HDFS;
end
```

Algorithm 3: Daily mapfile creation

4.4 Main Content Extraction

After crawling all the articles, the main task is to find a way to extract the main part of the news articles; we considered the following entities as the most important part of an article.

Article Fields

- **ID:** Article unique ID.
- **Headline:** Title of the article.
- **Content:** Main content of article.
- **Author:** The person who wrote the article.
- **Date:** The date of publication.
- **URL:** URL of the article.
- **Orientation:** Article biased label.
- **Publisher:** The news producer that published the article.

Several items of article fields' list are the information we already had at that stage such as ID, URL, Orientation, and Publisher. Moreover, the rest of the fields are information that we need to extract from the crawled news articles.

4.4.1 Writing the Wrappers

In this stage, we checked all the news producers that at least had one successful crawl and the crawl page also passed the archive quality check; the number amounts to 383 news producers. Each of these producers in the list has their specific web layout, and as a result, it was impossible to develop a general wrapper which is able to extract the required fields from this variety of layouts. Consequently, we were left with no choice other than going through the layout of each of these websites and writing their own special query selector to find required information for each field. the following code is an example of writing a wrapper for `http://leftvoice.org/` and its required query selectors to extract the information from the HTML article.

```
wrappers.addWrapper(new UriSpecificWrapper(
  // This wrapper will only be used for leftvoice.org
  // optional second parameter for path matching
  new UriPredicate("leftvoice.org"),
  // This wrapper
  new SelectorBasedWrapper()
    .withTag(Wrapper.TAG_TITLE, "article div.header-articulo h1")
    .withTag(Wrapper.TAG_AUTHOR, "article div.autor-articulo a")
    .withTag(Wrapper.TAG_DATE, "article div.row:nth-of-type(2)
        div.col-md-12 span")
    .withTag(Wrapper.TAG_CONTENT, new String[] {
      "article div.articulo p",
  })));
```

Implementation

We stored all the crawled news articles into mapfiles and as mentioned before we copy the mapfiles on HDFS. Accordingly, we found MapReduce[2008][8] as the most convenient programming model to implement our application. MapReduce is a programming paradigm which is designed to process a large amount of data on the cluster in parallel and distributed algorithm. Our implementation of content extraction gets a list of mapfiles as input, then it uses all mapfile files as an input of the mapper application. The mapper distributes the records of keys and values to executors. Each executer checks the key and if this record contains the "page.html" file, it starts to perform the content extraction. To do so, it finds the domain of the article, and it uses HTMLReader library to find the needed HTML elements by using domain and HTML content of the article. For all the elements including title and author we only keep the plain text without any extra and unnecessary HTML tags. Moreover, for content and date, we obey the following standard.

Content

Storing content needed more thought to realize which HTML tags are helpful and necessary to keep for the further analysis. Finally, we decided to keep paragraph, quote, and anchor elements from HTML code and remove all the

[8] `https://en.wikipedia.org/wiki/MapReduce`

other tags. In addition, for anchor elements, we considered extra criteria and we added an extra property to anchor as internal or external.

- **Internal:** When the anchor link pointed to the news producer that published the article, we consider that as internal and then we removed the "href" property from the anchor element.
- **External:** We consider an anchor element external when its link pointed to other websites. In this case, we also kept the "href" property for that element.

Date
We have a great variety of formats of date, while each of these news producers has their own specific way to display the publication date. Therefore, we required a method in order to recognize the date from text regardless of the format. Natty[9] is the library that we employed to extract the date from a text. Finally, we stored the data in ISO 8601[10] which 2000-02-20 is an example.

4.5 Corpus Formatting

We produce the corpus in two different formats JSON[11] lines and XML[12]. We mostly used the JSON line format for analysis which is described in more detail in section 5. Furthermore, XML format was also produced to present the corpus to participants of our task in SemEval 2019[13].

Field Description:

- **id:** Article id which is a number with 7 digits.
- **published-at:** A date in ISO 8601 format in which a news producer published the article.
- **title:** Headline of the article.
- **content:** The field content is used to store the main content of the article.

[9] http://natty.joestelmach.com/

[10] https://en.wikipedia.org/wiki/ISO_8601

[11] https://en.wikipedia.org/wiki/JSON

[12] https://en.wikipedia.org/wiki/XML

[13] https://pan.webis.de/semeval19/semeval19-web/

- **bias:** This field indicates the orientation of an article which can be "left", "left-center", "least", "right-center", or "right".
- **hyperpartisan:** Hyperpartisan is a boolean value. It is "true" when bias value is "left" or "right", otherwise the value of this field is "false".
- **url:** In this field the URL of the article is stored.
- **author:** Name of the author of the article can be found in this field.

4.5.1 JSON Line

Here, one can see the structure of an article and the way we stored in it in JSON format. In the end, the corpus contains several text files, and in each of these files, each line represents an article as a JSON object.

```
{
  "id":"0000001",
  "published-at":"YYYY-MM-DD",
  "title":"Headline",
  "content":"Main Content",
  "hyperpartisan":"true",
  "bias":"right",
  "url":"URL",
  "author":"Author"
}
```

4.5.2 XML

We used the XML format to produce the hyperpartisan corpus for our SemEval 2019 task(Kiesel et al.[2018]). For each article in a corpus, we produce a separate XML file as you can see a sample below. In addition, for each article, we add an article tag in ground truth file which contains the properties id, hyperpartisan, bias, url, and labeled-by. The field labeled-by is only used for SemEval task and is filled by value publisher for the all the articles that we have collected in this thesis.

Article Instance:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<articles>
  <article id="0000001" published-at="YYYY-MM-DD" title="HeadLine">
  Content
  </article>
</articles>
```

Ground Truth:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<articles>
  <article id='0000001' hyperpartisan='true' bias='right'
      url='URL' labeled-by='publisher'/>
  <article id='0000002' hyperpartisan='true' bias='least'
      url='URL' labeled-by='publisher'/>
</articles>
```

Chapter 5

Analysis

In this chapter, we focused on an analysis of the corpus. We begin with basic statistics about the articles, then we inspect the links that we found in the corpus and we extract some statistics about the anchor elements inside the corpus. The next step was to check the articles for text reuse. Finally, we tried several machine learning methods to predict news articles orientation and also biased from the unbiased news.

5.1 Corpus Statistics

In this stage of the thesis, we extract basic statistics about the number of the articles in general, in addition to the number of articles per each bias category in the corpus. Moreover, we also calculate article length and by that we mean how many words each article contains. Considering the length of the articles, we were wondering how many articles we might potentially lose if we consider a specific number for a minimum number of words per articles. Accordingly, we choose 4 different thresholds for this minimum length which are 2, 50, 100, and 150 words. According to the table 5.1 and 5.2, in total we have 2,282,423 news articles in the corpus. We then decided to stay with 50 words length as the minimum threshold for the article length since 50 words is a reasonable length for minimum length and also we do not lose so many articles. As a result, in the final corpus, we have in total 1,493,601 articles with the average length of 694 words, in which the share of the left category is 207,354, left-center 227,709, least bias 243,606, right-center 476,801, and 339,997 right wing articles.

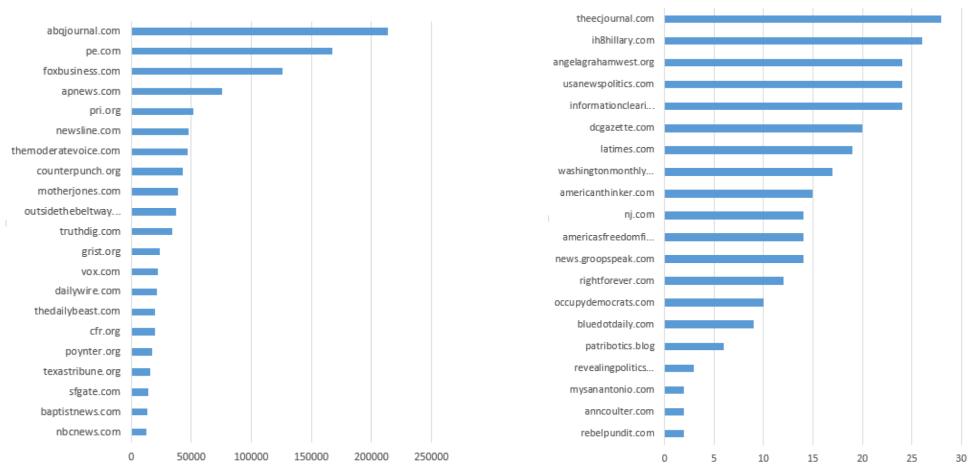
Figure 5.1 displays a list of top 20 producers that have the most and the least number of articles in our corpus.

Articles	left	Leftcenter	Least	Rightcenter	Right	Total
Total	311,511	837,422	551,673	335,552	442,461	2,478,620
Min. 2 words	262,345	228,467	459,324	272,239	362,164	1,583,173
Min. 50 words	253,311	214,598	423,675	262,078	334,413	1,488,075
Min. 100 words	243,863	203,245	396,570	238,297	313,334	1,395,309
Min. 150 words	232,717	189,475	365,429	212,733	290,031	1,289,785

Table 5.1: Corpus statistics about number of articles with different length

Average Length	left	Leftcenter	Least	Rightcenter	Right	Total
Min. 2 words	867	549	593	750	570	691
Min. 50 words	893	580	637	775	612	691
Min. 100 words	924	608	676	844	648	732
Min. 150 words	965	643	723	930	690	782

Table 5.2: Corpus statistics about average number of words in the articles



(a) Producers with the most articles

(b) Producer with the least articles

Figure 5.1: List of top 20 producers with the most and least articles

5.2 Corpus Anchor Element Analysis

In this section, we inspect the hyperpartisan corpus for anchor elements. As it was mentioned earlier in Corpus Construction Chapter 4, we categorized links into two categories, which are namely internal, and external. Internal links point to other articles from the same news producer and External links point to the URLs outside of the news producer. Table 5.3 demonstrates statistics about the number of links in the corpus in general, as well as the total number of links for each bias category. In total, we have 6,627,813 links and approximately 79 percent of all the links are external links. The other interesting fact is that on average, left category article have 7.45 links per article, the left-center ones have 5.72, least bias articles have 4.57, right-center have 3.04, and right bias articles have 4.21 links per article.

Article	left	Leftcenter	Least	Rightcenter	Right	Total
Total	1,545,640	1,230,138	967,071	1,453,404	1,431,560	6,627,813
Internal	368,951	341,425	289,446	141,971	280,257	1,422,050
External	1,176,689	888,713	677,625	1,311,433	1,151,303	5,205,763
Average	7.45	5.72	4.57	3.04	4.21	4.43

Table 5.3: Corpus anchor elements statistics

We also calculate the average number of the internal and external links per producer. Figure 5.2 shows the regarding statistics in the stacked bar chart in which the blue color represents the number of external links and the orange color shows the number of internal links per news producer.

Figure 5.3 displays the top 20 news producers in our corpus which are referenced in articles from other news producers. At the top of the list, we have the New York Times[1] and Washington Post[2] being referenced 135,591 and 97,001 times, both of which are left-wing news producers. Moreover, 7 news producers in this list are categorized as left bias, 6 left-center, 4 right, and 3 least bias category.

5.3 Duplicate Article Detection

The next obstacle of this thesis was to go through the corpus to examine the articles for similarity metrics. We wanted to be cognizant of how often

[1] http://nytimes.com

[2] http://washingtonpost.com

Figure 5.2: Average number of internal and external links per producer

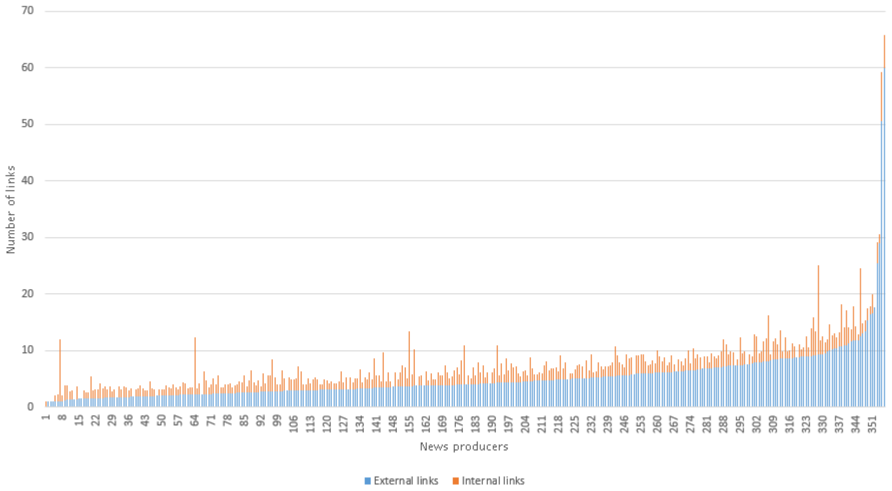
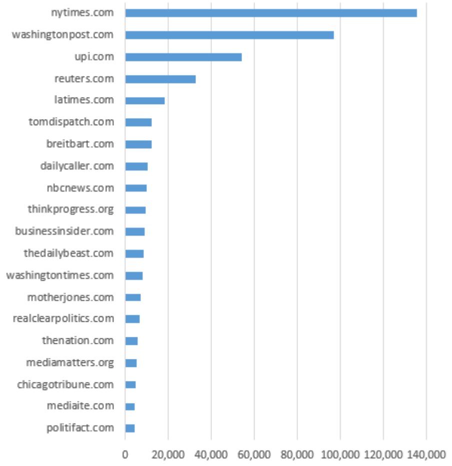


Figure 5.3: Top 20 news producers referenced by other producers

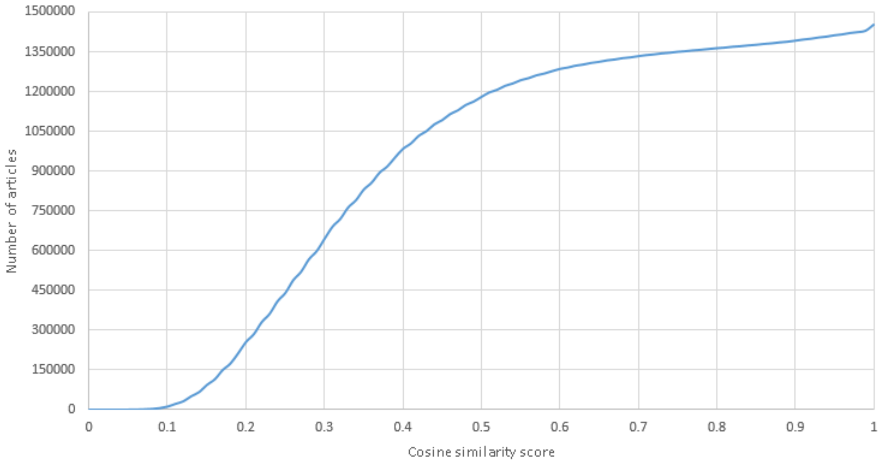


news producers reused the content from one another. Therefore, we focused on finding a way to recognize duplicate and near duplicate articles in the corpus. To overcome this problem, we employed a part of a pipeline that was introduced in Alshomary's thesis [2018] for text reuse analysis.

5.3.1 Text Reuse Analysis Pipeline

The actual goal of this pipeline is to find all possible pairs of text reuse from two collections of documents. This task has three subtasks including text preprocessing, candidate filtering, and text alignment. Text preprocessing subtask also has its steps, content extraction, text cleaning, and feature extraction. However, from the first subtask, we only need to use the feature extraction step. The second subtask computes the similarity measure and removes the pairs with low similarity. Computing the similarity measure is the only step that we used from the second subtask; consequently, we totally ignored the third one. Therefore, we used our hyperpartisan corpus as an input, then we employed a representation method based on TF-IDF weighting scheme to represent each document in the dataset in a set of feature vectors. Then, we utilize candidate similarity measure subtask, which uses cosine similarty[3] method to perform a pairwise similarity comparison on all the possible pair of articles. Figure 5.4 displays the statistics about the number of documents in the corpus that we can keep with different cosine similarity threshold from 0.0 to 1.0.

Figure 5.4: Number of article / cosine similarity score



After extracting pairwise similarity scores, we wanted to decide about which

[3] https://en.wikipedia.org/wiki/Cosine_similarity

similarity score is the most reasonable one to remove duplicate and near duplicate articles. To do so, we extract several pairs of articles with different similarity scores namely, 0.5, 0.75, 0.95, and 0.95. Next, we inspected each pair manually for text reuse. We then decided to stay with the score of 0.90. Furthermore, to remove one of the pairs in the corpus and keep the other one, we utilize a very simple method. The first priority is to keep the article which was published most recently, and in the cases that articles do not have a publishing date, we delete one randomly.

5.4 Classification Experiment

In this section, we concentrate on the other important objective of this thesis, which is developing a pipeline to detect both hyperpartisan news articles and their orientation as well. Accordingly, we developed a two different pipeline which gets our corpus as an input and one of them creates a model for hyperpartisan detection and the other one predicts articles orientation. The hyperpartisan detector is a binary classifier which predicts if an article is extremely biased or not, and the orientation detector enjoys a multi-class classifier which predicts the article bias category such as left, left-center, least, right-center, and right. For hyperpartisan detector, we balanced the number of articles, and we used 250K articles from each left and right wings, in addition to other bias categories we also include 167K articles each.

5.4.1 Logistic Regression

Logistic regression[4] is one the most used methods for binary classification which gives a discrete binary between 0 and 1 as an outcome. It measures the relationship between dependent variables(labels) and independent variables(features) and it estimates the probabilities using its regression function. In our experience, we follow these steps.

- Tokenizing the articles to the words.
- Removing stop words.
- We use the bag-of-words[5] model to represent the articles in the vector space.
- We randomly split the data to train and test data, which share of the train data is 70 percent.

[4] https://en.wikipedia.org/wiki/Logistic_regression

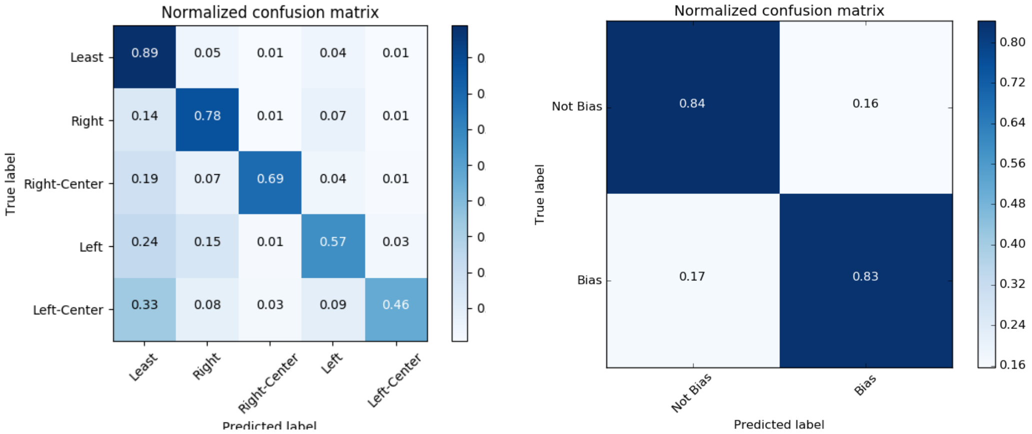
[5] https://en.wikipedia.org/wiki/Bag-of-words_model

- Finally, we trained the classifier using the logistic regression algorithm.

In this approach we produced two different classifiers, the first one is trained to predict the orientation of the algorithm, and the second one is used to detect partisan articles from least biased ones. Table 5.4 displays the performance of this algorithm over our corpus, and we were able to predict hyperpartisan article with a recall of 0.84, and also we predict the bias category of articles with a recall of 0.71.

	f1	Precision	Recall	Accuracy
Orientation	0.7077	0.7394	0.7126	0.7126
Hyperpartisan	0.8388	0.8385	0.8390	0.8398

Table 5.4: Logistic Regression outcome



(a) Bias category detection

(b) Hyperpartisan detection

Figure 5.5: Confusion matrix of Logistic Regression algorithm

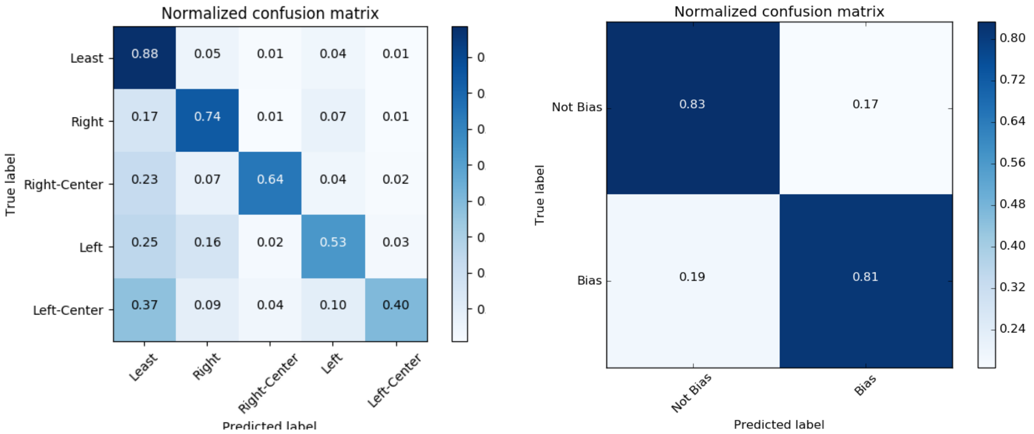
Method Two

This method also works similar to the previous one, However, in this experience, we used tf-idf[6] to convert articles to the vector space. Table 5.5 illustrates that using TF-IDF is not as successful as bag-of-words model. We achieved a recall of 0.82 for Hyperpartisan prediction and 0.68 to recognize the bias category of articles.

[6] https://en.wikipedia.org/wiki/Tf-idf

	f1	Precision	Recall	Accuracy
Orientation	0.6732	0.7110	0.6807	0.6807
Hyperpartisan	0.8239	0.8236	0.8231	0.8236

Table 5.5: Logistic Regression - TF-IDF



(a) Bias category detection

(b) Hyperpartisan detection

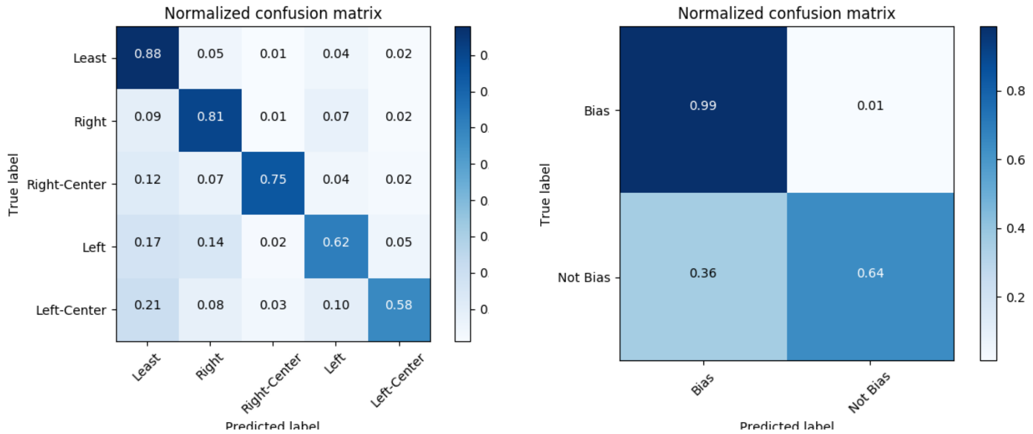
Figure 5.6: Confusion matrix of Logistic Regression algorithm

5-fold Cross Validation

Cross validation[7] is a method to increase the stability of the machine learning method. It tries to both find the correct pattern and remove the noises from the training data. K-fold cross-validation randomly splits the training data into K different subsets, then it trains the classifier K time, in each iteration, it considers one subset as the test data. Finally, the effectiveness of the model will be increased by averaging the error estimation of all K-folds. We also used this method in combination with our experience logistic regression algorithm and using the bag-of-words model, which increased the recall of hyperpartisan detection to 0.85 and 0.75 for the bias categorization. More detail can be seen in table 5.6.

	f1	Precision	Recall	Accuracy
Orientation	0.7495	0.7600	0.7522	0.7522
Hyperpartisan	0.8507	0.8498	0.8492	0.8508

Table 5.6: Logistic Regression with 5-fold cross validation



(a) Bias category detection

(b) Hyperpartisan detection

Figure 5.7: Confusion matrix of Logistic Regression with 5-fold cross validation

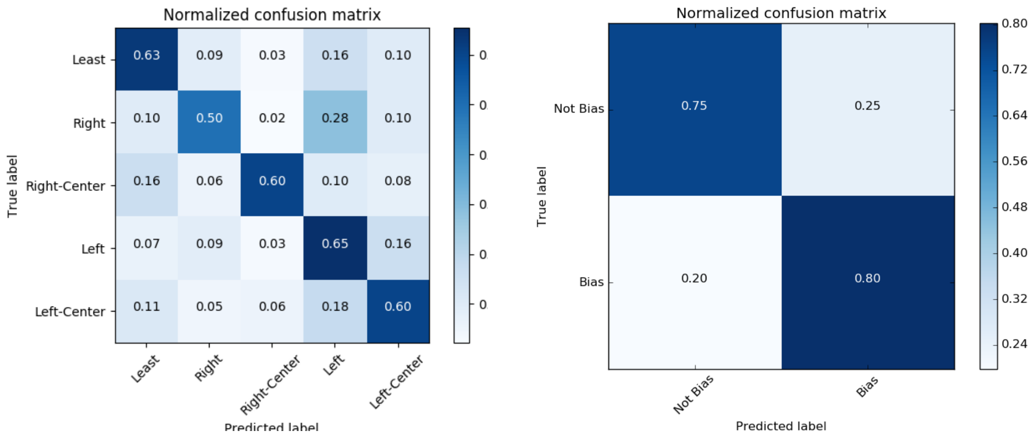
[7] https://en.wikipedia.org/wiki/Cross-validation_(statistics)

5.4.2 Naive Bayes

Naive Bayes[8] is a probabilistic classifier based on Bayes theorem which is very popular for the text classification. The fundamental rule of Naive Bayes is the independence of the features, and it presumes that the existence of a specific feature in a class is unrelated to the presence of any other feature. The steps that we took to prepare the data is similar to what we had for Logistic Regression method 5.4.1. First, we tokenize the articles to words, then we removed the stop words. The next step was to use the bag-of-words model to represent the articles in the vector space. Finally, we split the data into training and test set, and we trained the classifier using the Naive Bayes algorithm. Table 5.7 shows that we reached recall of 0.78 for hyperpartisan detection and 0.59 for orientation categorization.

	f1	Precision	Recall	Accuracy
Orientation	0.6023	0.6304	0.5954	0.5954
Hyperpartisan	0.7761	0.7784	0.7768	0.7776

Table 5.7: Naive Bayes algorithm outcome



(a) Bias category detection

(b) Hyperpartisan detection

Figure 5.8: Confusion matrix of Naive Bayes algorithm

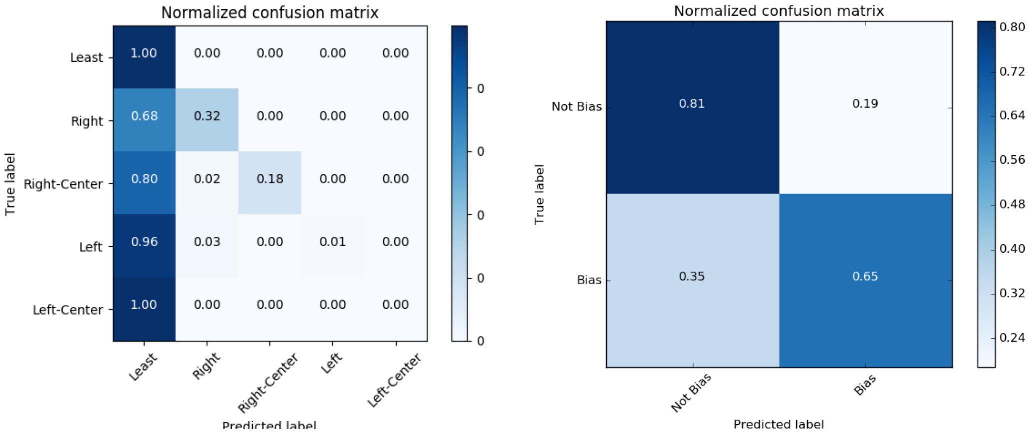
[8] https://en.wikipedia.org/wiki/Naive_Bayes_classifier

5.4.3 Random Forest Classifier

Random Forrest[9] is a supervised machine learning model, and it produces the model by merging several decision trees to increase the accuracy and stability of the model. We also utilize this algorithm in our study. Here also we followed the steps that we had in the Logistic Regression section 5.4.1. The outcome was not as impressive as our other experiments. The recall value that we achieved for hyperpartisan detection is 0.73 and 0.38 for the bias category detection which can be seen in more detail in table 5.8.

	f1	Precision	Recall	Accuracy
Orientation	0.3005	0.5681	0.3887	0.3887
Hyperpartisan	0.7306	0.7384	0.7328	0.7322

Table 5.8: Random Forest algorithm outcome



(a) Bias category detection

(b) Hyperpartisan detection

Figure 5.9: Confusion matrix of Random Forest algorithm

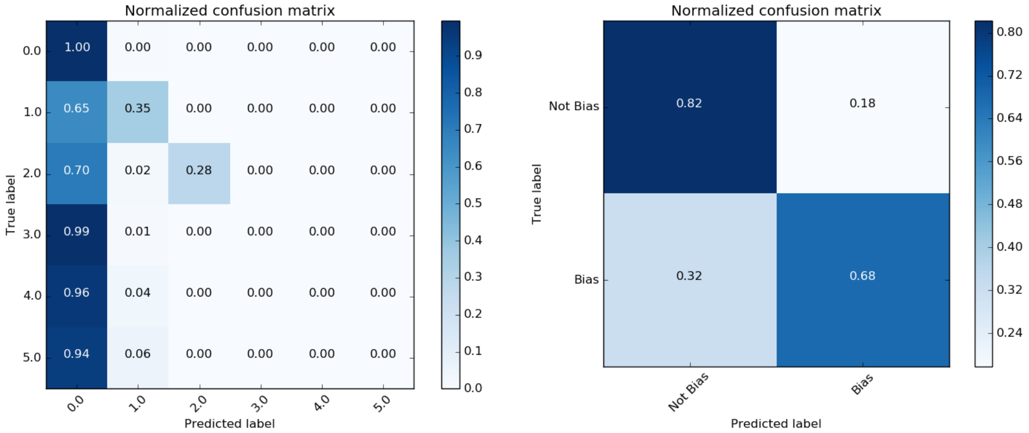
5-fold Cross Validation

In this experiment, we train the classifier using random forest and also with 5-fold validation method to make it more accurate. Table 5.9 demonstrates that the recall for hyperpartisan and bias category detection slightly increased to 0.75 and 0.41.

[9] https://en.wikipedia.org/wiki/Random_forest

	f1	Precision	Recall	Accuracy
Orientation	0.3300	0.6070	0.4114	0.4114
Hyperpartisan	0.7484	0.7559	0.7501	0.7501

Table 5.9: Random Forest with 5-fold cross validation



(a) Bias category detection

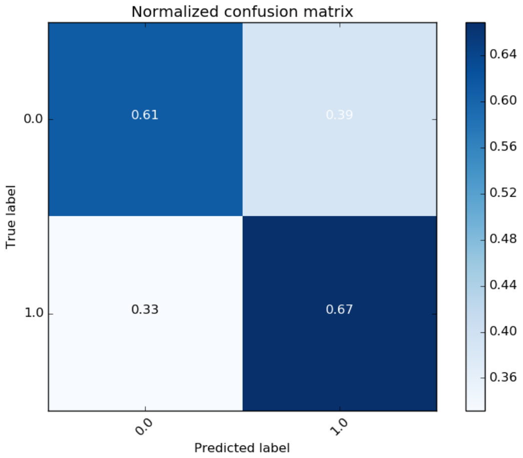
(b) Hyperpartisan detection

Figure 5.10: Confusion matrix of Random Forest with 5-fold cross validation

5.4.4 Experiment on SemEval 2019 Corpus

We produced this corpus for the participants in our SemEval 2019 Task. It contains 1 million articles, 800K articles for training and 200K articles for the test set. The training set has 200K left, 400K least, and 200K right articles, in addition, the test set has 50K left, 100K least, and 50K right articles. Moreover, we divided the data in the way that the training and test set share no news producers in common. Finally, the task is to predict the extreme bias of the news articles. We applied the most successful method that had in our experiment on this corpus, and consequently, we archive the recall of 0.64. The main reason that has this decrement in the accuracy is that in the previous experiment the classifier was trained over data from all the news producers. However, to increase the difficulty, in this corpus test set and training set have no news producer in common.

Figure 5.11: Confusion matrix of Logistic Regression with 5-fold cross validation on SemEval 2019 corpus



Chapter 6

Conclusion

In this study, we concentrate on a way to deal with ideological creation and contribution of political news. To do so, we felt a demand for a decent news articles dataset, and in order to overcome this issue, we create a hyperpartisan corpus as the first contribution of this thesis. To begin with, we found a list of news producers with their corresponding ideological agenda from two trustworthy sources namely, BuzzFeed, and Media Bias/Fact Check. In the next stage, we inspected Facebook pages and sitemaps of the websites of those news producers, which led us to extract all the links of their news articles. Then, we utilize the Betaweb cluster, in the combination of Webis Web Archiver tool to crawl all the links. In this phase, we develop an automatic tool which does the crawling process in combination with a quality check to guarantee the quality of the task. In the next step, we developed a wrapper to extract the main content from HTML pages, which is completely compatible with 358 news producers web layouts. Finally, we construct the corpus of 1.5 million articles in two different formats JSON lines and XML. The XML format was built exclusively for the participants of our SemEval 2019 task.

As the second contribution of this thesis, we analyze the corpus to answer several questions. First, we investigate hyperlinks in the articles, and the way news articles pointed to one another. Second, we perform a pairwise similarity check between all the possible pairs of articles in the corpus. This stage helped us to recognize duplicate and near duplicate articles in the corpus. Finally, we developed two pipelines, which utilize several machine learning methods to predict the article's orientation, and also hyperpartisanship. According to our best result that we achieved in our experiments, using Logistic Regression with 5-fold Cross Validation and our pipeline, we were able to distinguish orientation with ((F_1=0.74)) and hyperpartisanship of news articles with a remarkable score of ((F_1=0.85)).

Future Work

Building on our contributions in this thesis, there are still possibilities to enhance the quality of the corpus. One example could be further analysis of the data and topic classification in order to eliminate the non-political articles. In addition, there is always a chance to add new articles in the corpus. We can also take advantage of the statistics we collected to detect duplicate and near-duplicate documents, and we can go further in this regard to detect possible text reuse in the corpus. Considering our experiments on the data, we are looking forward to using another pipeline from Johannes Kiesel[2018], which owing to time limitation we were not able to utilize it. However, we adopt the implementation to be compatible with our corpus and it is quite interesting to figure out how it can perform with a big corpus, that we have constructed.

Bibliography

Milad Alshomary. A Pipeline for Scalable Text Reuse Analysis. Master's thesis, Bauhaus-Universität Weimar, Fakultät Medien, Computer Science and Media, July 2018. 5.3

E.L. Bernays and M.C. Miller. *Propaganda*. Ig Publishing, 2005. ISBN 9780970312594. URL `https://books.google.de/books?id=3De8nd_B_C8C`. (document)

Shweta Bhatt, Sagar Joglekar, Shehar Bano, and Nishanth Sastry. Illuminating an ecosystem of partisan websites. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, pages 545–554, Republic and Canton of Geneva, Switzerland, 2018. International World Wide Web Conferences Steering Committee. ISBN 978-1-4503-5640-4. doi: 10.1145/3184558.3188725. URL `https://doi.org/10.1145/3184558.3188725`. 2.2

Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, January 2008. ISSN 0001-0782. doi: 10.1145/1327452.1327492. URL `http://doi.acm.org/10.1145/1327452.1327492`. 4.4.1

Milad Alshomary Benno Stein Matthias Hagen Martin Potthast Johannes Kiesel, Florian Kneist. Reproducible web corpora: Interactive archiving with automatic quality assessment. *Journal of Data and Information Quality*, 2018. 4.1.1, 4.1.1, 4.1.3, 6

Johannes Kiesel, Martin Potthast, Maria Mestre, Rishabh Shukla, Benno Stein, David Corney, Emmanuel Vincent, and Payam Adineh. SemEval 2019 Task 4 - Hyperpartisan News Detection SemEval 2019 Task 4 - Hyperpartisan News Detection, July 2018. URL `https://doi.org/10.5281/zenodo.1400316`. 4.5.2

Lilliana Mason. A cross-cutting calmhow social sorting drives affective polarization. *Public Opinion Quarterly*, 80(S1):351–377, 2016. doi: 10.1093/poq/nfw001. URL `http://dx.doi.org/10.1093/poq/nfw001`. 1

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. A stylometric inquiry into hyperpartisan and fake news. *CoRR*, abs/1702.05638, 2017. URL `http://arxiv.org/abs/1702.05638`. 1, 2.3

Iyengar Shanto and Westwood Sean J. Fear and loathing across party lines: New evidence on group polarization. *American Journal of Political Science*, 59(3):690–707. doi: 10.1111/ajps.12152. URL `https://onlinelibrary.wiley.com/doi/abs/10.1111/ajps.12152`. 1

Tom White. *Hadoop: The Definitive Guide*. O'Reilly Media, Inc., 1st edition, 2009. ISBN 0596521979, 9780596521974. 4.2.1

Amy X. Zhang, Aditya Ranganathan, Sarah Emlen Metz, Scott Appling, Connie Moon Sehat, Norman Gilmore, Nick B. Adams, Emmanuel Vincent, Jennifer Lee, Martin Robbins, Ed Bice, Sandro Hawke, David Karger, and An Xiao Mina. A structured response to misinformation: Defining and annotating credibility indicators in news articles. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, pages 603–612, Republic and Canton of Geneva, Switzerland, 2018. International World Wide Web Conferences Steering Committee. ISBN 978-1-4503-5640-4. doi: 10.1145/3184558.3188731. URL `https://doi.org/10.1145/3184558.3188731`. 2.1