AMERICAN UNIVERSITY OF BEIRUT

DETECTION OF FAKE NEWS IN THE SYRIAN WAR

by
ROAA AL FEEL

A thesis
submitted in partial fulfillment of the requirements
for the degree of Master of Science
to the Department of Computer Science
of the Faculty of Arts and Sciences
at the American University of Beirut

Beirut, Lebanon
January 2019

AMERICAN UNIVERSITY OF BEIRUT

DETECTION OF FAKE NEWS IN THE SYRIAN WAR

by

ROAA AL FEEL

Committee Members:

	[illegible]
Dr. Fatima Abu Salem, Associate Professor	Advisor
Computer Science	
	[illegible]
Dr. Shady Elbassouni, Assistant Professor	Member of Committee
Computer Science	
	[illegible] on behalf of dr. Jaber
Dr. Mohamad Jaber, Assistant Professor	Member of Committee
Computer Science	
	[illegible]
Dr. May Farah, Assistant Professor	Member of Committee
Media Studies	

Date of thesis defense: January 28, 2019

AMERICAN UNIVERSITY OF BEIRUT

THESIS, DISSERTATION, PROJECT RELEASE FORM

Student Name:	Al Feel	Roaa	Bassem
	Last	First	Middle

[✓ Master's Thesis] ◯ Master's Project ◯ Doctoral Dissertation

[✓] I authorize the American University of Beirut to: (a) reproduce hard or electronic copies of my thesis, dissertation, or project; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes.

☐ I authorize the American University of Beirut, to: (a) reproduce hard or electronic copies of it; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes after: **One ___ year from the date of submission of my thesis, dissertation or project.**
Two ___ years from the date of submission of my thesis , dissertation or project.
Three ___ years from the date of submission of my thesis , dissertation or project.

[illegible]	05/02/2019
Signature	Date

This form is signed when submitting the thesis, dissertation, or project to the University Libraries

Acknowledgements

I would like to use this opportunity to express my gratitude to everyone who supported me throughout my masters study and through the process of researching and writing this thesis.

I would like to express my deep and sincere gratitude to my advisor Dr. Fatima Abu Salem for giving me the opportunity to be involved in this research project and her invaluable guidance, patience, motivation, enthusiasm, and immense knowledge throughout this research.

I am also grateful to the members of my thesis committee for all of their guidance throughout this process; your discussion, ideas, and feedback have been absolutely invaluable.

I would also like to thank my friends at AUB who have shared this experience with me for all the fun we have had in this journey.

And last but by no means least, I am extremely grateful to my parents and brother for all the love, support, and constant encouragement they have provided me with throughout my years of study.

This accomplishment would not have been possible without you all.

Thank you.

An Abstract of the Thesis of

Roaa Al Feel for Master of Science
Major: Computer Science

Title: Detection of Fake News in the Syrian War

After almost eight years of conflict, the humanitarian situation in Syria continues to deteriorate year after year. With multiple opposing parties involved in the armed conflict, much of the news reported about the Syrian war seems to be biased or inclined to support a certain party over the others. With serious human rights violations taking place in the Syrian war, and news sources blaming different sides of the conflict for these violations, interest in the detection of fake news surrounds the Syrian war.
In this work, we built a streaming and scraping model to extract news articles of interest from news sources' websites. We built a labeled dataset of news articles about the Syrian conflict. Finally, we built a feature extraction model along with a machine learning model that is able to detect fake news in the Syrian conflict and generalize to other types of fake news.

Contents

Acknowledgements **v**

Abstract **vi**

List of Figures **xi**

List of Tables **xiii**

1 Introduction **1**

2 Literature Review **2**

2.1 FakeNewsNet: A Data Repository with News Content, Social Context and Dynamic Information for Studying Fake News on Social Media . . . 2

2.1.1 Comparison to Our Approach . . . 3

2.2 The Data Challenge in Misinformation Detection: Source Reputation vs. Content Veracity . . . 3

2.2.1 Comparison to Our Approach . . . 3

2.3 Fake News vs Satire: A Dataset and Analysis . . . 4

2.3.1 Comparison to Our Approach . . . 4

2.4 Liar, Liar Pants on Fire: A New Benchmark Dataset for Fake News Detection . . . 4

2.4.1 Comparison to Our Approach . . . 5

2.5 ClaimBuster . . . 5

2.5.1 Comparison to Our Approach . . . 5

2.6 Leveraging Joint Interactions for Credibility Analysis in News Communities *AND* Credibility Assessment of Textual Claims on the Web 6

2.6.1 Comparison to Our Work . . . 6

3 Building The Dataset of Articles **7**

3.1 The StorageHandler Interface . . . 7

3.2 The Streamer Abstract Class . . . 8

3.3 The Scraper Interface . . . 8

3.4 The TwitterStreamer Class . . . 8

3.4.1 TwitterStreamer's Streaming Functions 8
3.5 URLStreamerScraper 9
3.5.1 The Streaming Functions 10
3.5.2 The Scraping Functions 10
3.6 NewsReceiver 10
3.7 WebpageScraper 11
3.8 Articles Scraped 11
3.8.1 Sources Scraped 11
3.8.2 Events Scraped 12

4 Labeling The Dataset 14
4.1 VDC 14
4.2 Extracting Article Claims 15
4.2.1 Figure-Eight Job 16
4.2.2 Manual Quality Control 20
4.3 Mapping Article Claims to the VDC Casualties 22
4.3.1 Aggregating the VDC Casualties into Events 22
4.3.2 Mapping Each Article to a VDC Event 23
4.4 Clustering The Articles 24
4.4.1 Setting The Clustering Features 24
4.5 Experiments and Results 25
4.5.1 Examples 26

5 Feature Extraction 32
5.1 Stats About the Label 32
5.1.1 Stats About the Events and Dates 32
5.1.2 Stats About the Sources 34
5.2 List of Features Extracted 37
5.3 Sectarian Language 42
5.3.1 Examples of Sectarian Words in Our Dataset 43
5.3.2 Extracting the Feature 44
5.3.3 Exploratory Analysis 44
5.4 Consistency Score 44
5.4.1 Extracting the Feature 46
5.4.2 Exploratory Analysis 47
5.5 Description of Sources Quoted in Reporting the News 48
5.5.1 Extracting the Feature 49
5.5.2 Exploratory Analysis 49
5.6 Assertive Verbs 49
5.6.1 Examples of Assertive Verbs in Our Dataset 51
5.6.2 Extracting the Feature 52
5.6.3 Exploratory Analysis 52
5.7 Factive verbs 54

5.7.1 Examples of Factive Verbs in Our Dataset 54
5.7.2 Extracting the Feature 55
5.7.3 Exploratory Analysis 55
5.8 Hedges 55
5.8.1 Examples of Hedges in Our Dataset 57
5.8.2 Extracting the Feature 58
5.8.3 Exploratory Analysis 58
5.8.4 Hedges 58
5.9 Implicative Verbs 59
5.9.1 Examples of Implicative Verbs in Our Dataset 59
5.9.2 Extracting the Feature 60
5.9.3 Exploratory Analysis 61
5.10 Report Verbs 61
5.10.1 Examples of Report Verbs in Our Dataset 63
5.10.2 Extracting the Feature 64
5.10.3 Exploratory Analysis 64
5.10.4 Report Verbs 64
5.11 Subjectivity and Bias 65
5.11.1 Examples of Bias in Our Dataset 65
5.11.2 Extracting the Feature 66
5.11.3 Exploratory Analysis 68
5.12 Learning Curve 68

6 Methodology **71**
6.1 Baseline Models 71
6.1.1 Ripper Rules 71
6.1.2 See5 Rules 72
6.2 Machine Learning Model 73
6.2.1 SVM 74
6.2.2 Decision Tree Classifier 74
6.2.3 Polynomial Linear Regression 74
6.2.4 Naive Bayes 74
6.3 Feature Selection 74
6.3.1 SelectKBest 74
6.3.2 Feature Importance 75

7 Results **77**
7.1 Best Model 77
7.2 Testing Model on Related Work Dataset 77
7.3 Testing The Quality of Our Dataset 77

8 Discussion **81**
8.1 News Streaming Scraping Module 81
8.2 Fake News Detection Model 81
8.3 Factcheck.org 82
8.3.1 Example 1: Trump Claims New York Times Article Is Fake 82
8.3.2 Example 2: A Story Goes Viral About McCain Following His Death 83

9 Conclusion **84**

Bibliography **85**

List of Figures

4.1 Screenshot of the Crowd Answers 18
4.2 Screenshot of the Data Extracted from VDC 23
4.3 Screenshot of the VDC Events After Aggregation 23
4.4 Figure showing the silhouette coefficient of the clusters for each window size 29
4.5 Figure showing the PCA plot of the clusters 30
4.6 Plot Showing The Number of Fake Articles Where A Clustering Feature Was Different From VDC 31

5.1 Plot Showing the number of articles labeled true and number of articles labeled fake independent of the source or category. 33
5.2 Plot Showing the % of Articles Labeled True for Each Event where % = nb_true_per_event / nb_total_per_event 35
5.3 Plot Showing the % of Articles Labeled Fake for Each Event where % = nb_fake_per_event / nb_total_per_event 36
5.4 Plot Showing the Number of Articles Labeled Fake For Each Month 37
5.5 Plot Showing the % of Articles Labeled True for Each Category where % = nb_true_per_category / nb_total_per_category 38
5.6 Plots Showing the % of Articles Labeled True/Fake for Each Source Category (Pro, Against, Neutral) 39
5.7 Plot Showing the % of Articles Labeled True for Each Source where % = nb_true_per_source / nb_total_per_source 40
5.8 Plot Showing the % of Articles Labeled Fake for Each Source where % = nb_fake_per_source / nb_total_per_source 41
5.9 Plot Showing the Distribution of Sectarian Language Frequencies in Articles Labeled Fake 45
5.10 Plot Showing the Distribution of Sectarian Language Frequencies in Articles Labeled True 45
5.11 Plot Showing the Distribution of the Consistency Scores in Articles Labeled Fake 47
5.12 Plot Showing the Distribution of the Consistency Scores in Articles Labeled True 48

5.13 Plot Showing the Distribution of the Quoted Sources Feature in Articles Labeled Fake 50
5.14 Plot Showing the Distribution of the Quoted Sources Feature in Articles Labeled True 50
5.15 Plot Showing the Distribution of Assertive Verbs Frequencies in Articles Labeled Fake 52
5.16 Plot Showing the Distribution of Assertive Verbs Frequencies in Articles Labeled True 53
5.17 Plot Showing the Distribution of Factive Verbs Frequencies in Articles Labeled Fake 56
5.18 Plot Showing the Distribution of Factive Verbs Frequencies in Articles Labeled True 56
5.19 Plot Showing the Distribution of Hedges Frequencies in Articles Labeled Fake 58
5.20 Plot Showing the Distribution of Hedges Frequencies in Articles Labeled True 59
5.21 Plot Showing the Distribution of Implicative Verbs Frequencies in Articles Labeled Fake 61
5.22 Plot Showing the Distribution of Implicative Verbs Frequencies in Articles Labeled True 62
5.23 Plot Showing the Distribution of Report Verbs Frequencies in Articles Labeled Fake 64
5.24 Plot Showing the Distribution of Report Verbs Frequencies in Articles Labeled True 65
5.25 Plot Showing the Distribution of Bias Frequencies in Articles Labeled Fake 67
5.26 Plot Showing the Distribution of Bias Frequencies in Articles Labeled True 67
5.27 Learning Curve Using a Linear Regression Model 68
5.28 Plot Showing the Correlation HeatMap Of The Features 70

6.1 Feature Importance 76

7.1 ROC Curve for SVM Model 79
7.2 ROC Curve for Decision Tree Classifier 79
7.3 ROC Curve for Polynomial Linear Regression 80
7.4 ROC Curve for Naive Bayes 80

List of Tables

3.1 Major Events in the Syrian War Extracted From VDC 13

4.1 Fleiss Kappa Agreement of the Contributors for Each Question . 19

7.1 Results of Tested Machine Learning Models 78
7.2 Confusion Matrix for the SVM Model 78
7.3 Confusion Matrix for the Decision Tree Classifier 78
7.4 Confusion Matrix for the Polynomial Linear Regression Model . . 78
7.5 Confusion Matrix for the Naives Bayes Model 78

Chapter 1

Introduction

After almost eight years of conflict, the humanitarian situation in Syria continues to deteriorate year after year. With multiple opposing parties involved in the armed conflict, much of the news reported about the Syrian war seems to be biased or inclined to support a certain party over the others. With serious human rights violations taking place in the Syrian war, and news sources blaming different sides of the conflict for these violations, interest in the detection of fake news surrounds the Syrian war.
In this work, we built a streaming and scraping module to extract news articles of interest from news sources' websites. In order to achieve this, we first built a model that works on top of Spark and Spark streaming that allows us to scrape news sources, such as newspaper websites, for content published regarding a certain event or conflict. Our model can also stream the news source for new content regarding the event of interest, and store any streamed or scraped content in HBase.
We then used this model to build a dataset of news articles regarding the Syrian conflict and label them as fake or credible. We achieved this by extracting facts from the articles through crowdsourcing. We then compared these facts against the Violations Documentation Center (VDC), which is a neutral humanitarian organization that has been documenting the events of the Syrian war since 2011. We then clustered the articles into fake and credible clusters.
Finally, we performed feature-extraction on our dataset and trained and tested different machine learning models to find the one that gives the best results for our problem. We also tested the best found model on other fake news datasets to prove that our model generalizes to other types of fake news and not just the Syrian war.

Chapter 2

Literature Review

In this chapter, we take a look at some of the works that have been done to solve the fake news detection problem.

2.1 FakeNewsNet: A Data Repository with News Content, Social Context and Dynamic Information for Studying Fake News on Social Media

In [1], the authors build a dataset of news labeled true and fake. In order to build this labeled dataset, they build crawlers that crawl fact-checking websites such as Politificat (for political news) and GossipCop (for entertainment news) to obtain news content for fake news and true news. Both of these websites provide analysis done by journalists and domain experts to label news articles as fake and real. The authors also crawl E! online for entertainment news pieces and consider all their news as real as they believe this source is a trusted source.

The authors then perform data analysis to the dataset they have built and take a look at features such as:

- News Content: These include the meta information related to a piece of news such as publishers, headlines, body texts, and images/videos.
- User Profile: These include the creation time of user accounts, whether the user is a bot or not, etc..
- User Posts: The authors perform sentiment analysis on the replies of user posts spreading the real and fake news.
- Network Structures: The authors take a look at features such as followers and followee count, likes, retweets, and replies of the tweets.

- Dynamic Information: The authors take a look at temporal behaviors of users spreading fake news and real news such as the posting time and day of the week.

The authors then use the above features and fit their dataset on several machine learning models (e.g. SVM, Naive Bayes, CNN, etc...) in order to perform the fake news detection task.

2.1.1 Comparison to Our Approach

The authors of [1] rely on journalists and domain experts' analysis and reviews to label their dataset. In addition, they trust that all articles from E! are true since they trust this source. However, we use clustering in our work to automate the labeling of the dataset and do not label articles based on whether we believe their source is trusted or not. This allows us to prevent human bias from affecting our labels. In addition, the features discussed by [1] consist mostly of the meta-data about the news article rather than the content itself. In our approach, we extract features from the articles themselves for our feature extraction.

2.2 The Data Challenge in Misinformation Detection: Source Reputation vs. Content Veracity

In [2], the authors introduce two datasets scraped from the web by leveraging links to news articles mentioned by fact-checking websites (Buzzfeed and Snopes) with their labels. Both datasets used by [2] are made up of full articles labeled by human experts. These labels were: true, mostly true, mixture of true and false, mostly false, and false stories out of this website.
The authors then use TF-IDF, n-grams, word vector, etc... in order to perform text classification for the fake news detection problem using their datasets.

2.2.1 Comparison to Our Approach

Similar to the works of [2], we rely on the actual content of the news articles rather than the reputation of the sources. However, the authors of [2] do not extract features and rely on n-grams, TF-IDF, and similar text-based techniques. This approach could allow the writing style of the authors of the articles to be a factor in the classification. In order to prevent this, we rely on features extracted from the articles rather than the text of the article itself. In addition, the authors rely on human experts to label their dataset, which could be subject to human bias. In our approach, we avoid this by automating the labeling of the articles based on the facts provided by the article.

2.3 Fake News vs Satire: A Dataset and Analysis

The authors of [3] built a dataset of fake news and satirical stories and restricted their dataset to American politics, recent articles, diverse sources, and no borderline cases. They identified a list of fake news and satirical websites and assigned researchers for each website to label each article scraped from the website as fake or satirical. The authors then build a Naive Bayes Multinomial model to classify an article based only on the language it used. Each article was represented as a word vector with a class of Fake or Satire.
The authors then explore the themes of the articles by taking a look at the types of content that are shared. They manually labeled each article in the dataset as one or more of: hyperbolic position against one person or group, discredit a normally credible source, racist messaging, conspiracy theory etc...
The authors then use these themes as an additional feature to the word vector dataset to see if they make a difference in the results. However, they found no significant difference in the accuracy or AUC.

2.3.1 Comparison to Our Approach

The differences and similarities between our approach and the approaches of [2] and [3] are the same. Both [2] and [3] rely on the actual content of the news articles rather than the reputation of the sources, which is what we do in our approach. However, both [2] and [3] do not extract features and rely only on the language used. They use text-based techniques such as word vectors, n-grams, TF-IDF, etc... which we avoid in our approach and rely features extracted from the articles rather than the text of the article itself. In addition, the authors of [3] also rely on human experts to label their dataset, which could be subject to human bias. In our approach, we avoid this by automating the labeling of the articles based on the facts provided by the article.

2.4 Liar, Liar Pants on Fire: A New Benchmark Dataset for Fake News Detection

The authors of [4] built the LIAR dataset, which includes 12.8K human labeled short statements from politifcat.com's API. They consider six fine-grained labels for each statement: pants-fire, false, barely-true, half-true, mostly-true, and true. These statements were sampled from news releases, TV/radio interviews, campaign speeches, tweets, etc.. The subjects of these tweets include economy, health-care, taxes, education, jobs, elections, etc...
Once the authors built this dataset, they built models to test the fake news detec-

tion problem using this dataset. They used models such as a majority baseline, a regularized logistic regressin classifier, SVM, etc... to perform a 6-way multiclass text classification for fake news detection.

2.4.1 Comparison to Our Approach

The dataset of [4] consists of labeled statements rather than full articles. In our approach, we work with full news articles instead. In addition, the features [4] use are word vectors. In our approach, we use features extracted from the articles rather than word vectors.

2.5 ClaimBuster

In [5] the authors build a system called "Claim" Buster that interfaces data sources, identifies factual claims in verbose texts from these sources, matches them with existing fact-checks that are related to the discovered claims, and displays the report through for the users. The "Claim Spotter" module of the "Claim Buster" is a classification and scoring model that was trained using sentences from past general election debates. The "Claim Matcher" module of the "Claim Buster" searches a fact-check repository and returns those fact-checks matching the claim. This repository was curated from fact-checking websites. The "Claim Checker" module of the "Claim Buster" collects supporting or debunking evidence from knowledge bases and the web. If any clear discrepancies between the returned answers the claim exist, a verdict is derived and presented to the users. Finally, the "Fact-check Reporter" module of the system synthesizes a report combing the evidence and delivers it to the users through the "Claim Buster"'s website.

2.5.1 Comparison to Our Approach

The authors of [5] rely on fact-checking websites and automated Google searches to label a given claim, which could be subject to the bias of the fact-checking websites scraped and that of the articles returned by the Google search. However, we use clustering in our work to automate the labeling of the dataset which allows us to prevent human bias from affecting our labels.

2.6 Leveraging Joint Interactions for Credibility Analysis in News Communities *AND* Credibility Assessment of Textual Claims on the Web

In both [6] and [7], the authors use a set of lexicons to extract the frequencies of the following words in each article of their dataset:

- Assertive verbs (e.g. claim): capture the degree of certainty to which a proposition holds.
- Factive verbs (e.g. indicate): presuppose the truth of a proposition in a sentence.
- Hedges (e.g. may): soften the degree of commitment to a proposition.
- Implicatives: trigger presupposition in an utterance (e.g. usage of the word complicit indicates participation in an activity in an unlawful way).
- Report verbs (e.g. argue): emphasize the attitude towards the source of the information.
- Subjectivity and bias: News is supposed to be objective, writers should not convey their own opinions, feelings or prejudices in their stories. The authors use a subjectivity lexicon, a list of positive and negative opinionated words, and an affective lexicon to detect subjective clues in articles.

However, the authors in [7] add to above a feature called source reliability. According to the authors, apart from the reporting style of the article, the reliability of the web-source hosting the article also has a significant impact on the credibility of the claim. To capture the reliability of the web-source for each web article, they used PageRank, which determines importance of the website by counting the number and quality of links to and from the website.

2.6.1 Comparison to Our Work

Similar to [6] and [7], we use features extracted from the articles rather than relying on word vectors, therefore avoiding allowing the style of writing affecting our model. We use the above features suggested by both [6] and [7] and add a set of features to them as well. However, unlike [7], we do not use the sources of the articles as a feature in order to prevent bias against certain sources vs others.

Chapter 3

Building The Dataset of Articles

In order to build our dataset of Syria news articles, we implemented a general module that works on top of **Spark** and **Spark Streaming** to scrape and stream news. The **News Streaming and Scraping** module works on top of Spark Streaming to stream a given source for possible updates, or scrape the source for old content. The module consists of a **Streamer** abstract class which we extend in two different classes: the **URLStreamerScraper** class and the **TwitterStreamer** class. The **URLStreamerScraper** class streams a given website to detect new articles that contain a set of keywords, or scrapes the website for old articles that contain the set of keywords, whereas the **TwitterStreamer** class streams Twitter for tweets related to a given set of keywords. In both classes, the streamer periodically checks for updates from the source and saves them into the chosen form of storage. In this section, we describe the structure and functionalities of each of the classes provided in this module.

3.1 The StorageHandler Interface

The **StorageHandler** interface is required by the **Streamer** class in order to store the streamed news in a form of storage such as **HBase**. It consists of the following abstract functions:

- *initStorageStructure()*: a function that creates the storage structure such as creating tables, files, etc... This function depends on the storage software of choice.

- *saveNews(JavaDStream<? extends News> newsStream)*: a function that saves a stream of **News** to the chosen storage software.

- *saveNews(JavaRDD<? extends News> newsRDD)*: a function that saves a Spark RDD of **News** to the chosen storage software.

- *exportLocalNewsCSV(String fileName)*: a function that exports the **News** items stored in the storage software to a CSV file.
- *exportLocalNewsTXT(String directory)*: a function that exports the **News** items stored in the storage software to a directory, where each news item is exported as a TXT file.

3.2 The Streamer Abstract Class

The **Streamer** abstract class is the superclass which our streaming classes extend in order to stream a given source. It consists of a JavaSparkContext instance, a **StorageHandler** instance, setters and getters, and an abstract stream function: *stream(String[] keywords, int k).* This abstract function should be implemented in a class that extends the **Streamer** class to stream the required source for a given set of keywords every k seconds, minutes, or hours, etc...

3.3 The Scraper Interface

The **Scraper** interface is the interface which our scraping class implements in order to connect once to a given source and scrape it for old content. It consists of two abstract scraping functions: *scrape(String[] keywords)*, which should connect to a given source to scrape it for old content that contains the required set of keywords, and *scrape(String[] keywords, String[] years, String[] months)*, which should filter out scraped content that was not posted in the required year(s) and month(s). These abstract functions should be implemented in a class that implements the **Scraper** interface.

3.4 The TwitterStreamer Class

The **TwitterStreamer** class streams Twitter for new tweets related to a given set of keywords. It allows to either retrieve new tweets from a specified set of Twitter users, or from any Twitter user, as long as the tweet contains at least one of the required keywords.

3.4.1 TwitterStreamer's Streaming Functions

The **TwitterStreamer** class extends the **Streamer** abstract class and overrides its *stream(String[] keywords, int k)* function to stream Twitter for tweets related to a given set of keywords. In addition, this class has two streaming functions: the *streamTwitterFromSources(List<String> sources, String[] keywords, int minutes)* function and the *streamTwitter(String[] keywords, int minutes)* function.

Both functions call the class's overridden function *stream(String[] keywords, int k)* to stream Twitter for new tweets related to a given set of keywords.

The Overridden Stream Function

The **TwitterStreamer** class overrides the **Streamer** abstract class function *stream(String[] keywords, int k)*, discussed earlier, in order to stream Twitter every k minutes.

We first start by creating a JavaStreamingContext instance that connects to Twitter every k minutes. We then create a receiver, an instance of Spark Streaming's JavaReceiverInputDStream, which will receive the required stream of tweets every k minutes. We achieve this by making use of Spark Streaming's function: TwitterUtil.createStream. It creates an input stream that returns a continuous stream of tweets received from Twitter using Spark's JavaDStream. Using Spark's JavaDStream allows us to perform Spark's transformations such as map and filter on the stream of tweets retrieved, which guarantees parallelization and fault-tolerance. In case we want to retrieve the tweets posted by a certain set of Twitter users, we then apply Spark's filter transformation on these tweets to keep only the tweets posted by the Twitter users required. Using Spark's functions such as map and filter to fetch the tweets and keep the relevant ones allows us to parallelize the streaming process.

The **TwitterStreamer** now periodically connects to Twitter API, fetches the needed tweets, prints them to the console, and stores them in the required storage.

3.5 URLStreamerScraper

The **URLStreamerScraper** class extends the **Streamer** abstract class and overrides its *stream(String[] keywords, int k)* function to stream a given news website for new articles related to a given set of keywords. In addition, it implements the Scraper interface and overrides its *scrape(String[] keywords)* function to scrape the news website for articles previously published related to a given set of keywords. It consists of four main functions: streamFromHomePage(String[] articleKeywords, String homepageURL, int minutes, int maxDepth), streamFromSubURL(String[] articleKeywords, String homepageURL, String subURL, int minutes, int maxDepth), scrapeFromHomePage(String[] articleKeywords, String homepageURL, int maxDepth), and scrapeFromSubURL(String[] articleKeywords, String homepageURL, String subURL, int maxDepth).
In all four main functions of the **URLStreamerScraper**, the *articleKeywords* array corresponds to the set of keywords about the event of interest. The article

should contain at least one of these keywords to be considered relevant to the event of interest.
For example, in case we are scraping Al Arabiya's news website, the homepageURL is Al Arabiya's homepage. The subURL of the *streamFromSubURL()* and *scrapeFromSubURL()* functions correspond to a particular page of interest in the source's website, e.g. the link to the archive section in Al Arabiya's website. In case the from home page functions are called, the streaming/scraping begins from the homepageURL, whereas calling the subURL functions starts the streaming/scraping from the subURL, which is useful in case the website provides links to news archives, a section dedicated to the Syrian war, etc... The maxDepth parameter corresponds to the maximum number of links the streaming/scraping function visits before returning. In case of streaming, the function re-streams from the beginning (either the homepage or sub-url) after the specified number of minutes.

3.5.1 The Streaming Functions

Both *streamFromHomePage()* and *streamFromSubURL()* functions call the overridden *stream()* function. The overridden *stream()* function works in a similar manner to the **TwitterStreamer**'s stream function, by connecting periodically to a custom-created Java **NewsReceiver** class, discussed below. It creates a input stream that returns a continuous stream of **News** articles received from the **NewsReceiver** using Spark's JavaDStream. The URLStreamer now periodically connects to the **NewsReceiver** class, prints the streamed **NewsArticles** to the console, and stores them in the required storage.

3.5.2 The Scraping Functions

Both *scrapeFromHomePage()* and *scrapeFromSubURL()* functions call the overridden *scrape(String[] keywords)* function. The overridden scrape function connects once to the source, fetches all the articles related to the given set of keywords, saves them in the storage of choice, prints them to the screen, and returns. This is achieved by creating an instance of the **WebpageScraper** (discussed below) and calling its scrape function, which handles the scraping.

3.6 NewsReceiver

When streaming a given website for updates, the URLStreamer periodically connects to the **NewsReceiver** class which extends Spark's Receiver class that allows the storage of the retrieved **News** articles. That is, the **NewsReceiver** class will be started periodically, and restarted after the end of each streaming period to ensure continuous checking of new articles from the required source related to the required set of keywords.

Once started (or restarted) the **NewsReceiver** class calls the **Webpage-Scraper**'s scrape function that scrapes the webpage looking for articles related to the keywords the user is looking for.

3.7 WebpageScraper

The **WebpageScraper** does the whole parsing/scraping part of the streaming process. The **WebpageScraper** first parses the homeURL set by the user, looking for links (either strictly those related to the given set of keywords, or any link found, depending on the user's preference). Then, it looks in each of the required links in this webpage, parses it, and checks its links as well, up to the set depth of links set by the user.

The link is considered related to the required set of keywords if it contains any of the required keywords either in its URL or in the title of the link. In this case, the **NewsReceiver** will store the entire content of the webpage this link refers to.

In order to make sure we retrieve the entire content of all webpages, including those that rely on Javascript and AJAX calls to load entirely, we used the HtmlUnit library and the NicelyResynchronizingAjaxController to parse the webpages, which works as if we are connecting to the URL using a Javascript-enabled browser, thus allowing us to retrieve the entire content of each webpage we connect to, and then extract the needed links and articles from them.

Since we are periodically checking the website for updates related to the topic, we could be subject to connecting to the same webpage twice, which could be unnecessary if the webpage does not contain any new stories, links, or content, since nothing was added to these pages yet. Thus, to avoid unnecessary trials to each link we encounter, we send the 'if-modified-since' header to check if the webpage was modified since the last trial. If not, the website will respond with a '304 not modified' and there is no need to connect to the webpage.

3.8 Articles Scraped

In this section, we describe the sources scraped/streamed for articles reporting about some key events in the Syrian war.

3.8.1 Sources Scraped

In order to make sure that neither our dataset nor our model is biased towards or against one party or opinion in the Syrian war, we made sure to scrape for articles about the war events in the Syrian war from all sides/perspectives. The sources we scraped articles from were Turkish, Arab, Syrian, Russian, Iranian, and international, therefore ensuring that we have in our dataset all sides and opinions regarding the events in the Syrian war. The following is the list of sources that we scraped for:

- Al Arabiya : Saudi news channel. Against Syrian Regime.
- Jordan Times: Jordan Newspaper. Against Syrian Regime.
- Al Ahram: Egyptian Newspaper. Against Syrian Regime.
- Asharq Alawsat: Lebanese Newspaper. Against Syrian Regime.
- SANA: Syrian news channel. Pro-Syrian Regime.
- Al Alam: Iranian news channel broadcasting in Arabic. Pro-Syrian Regime.
- Al Manar: Lebanese news channel associated with Hizbollah. Pro-Syrian Regime.
- Sputnik: Russian news channel. Pro-Syrian Regime.
- TASS: Russian news channel. Pro-Syrian Regime.
- Reuters: International news agency headquartered in the UK. Libertarian/Neutral.
- Etilaf.org: The official website of the Syrian National Coalition Of Syrian Revolution and Opposition Forces. Against Syrian Regime.
- Al Araby: An online news website first launched in the UK. Against Syrian Regime.
- TRT World: Turkish news channel broadcasting in English. Against Syrian Regime.
- Daily Sabah: Turkish newspaper. Against Syrian Regime.

3.8.2 Events Scraped

In order to make sure to include as many major and controversial war events from the Syrian war as possible, we took a look at the peaks in the casualties reported in the VDC. These peaks were either peaks in a certain month (e.g. a sudden increase in the deaths by chemical weapons in Aleppo in August 2016), or long periods of similar events, but not necessarily peaks (e.g. deaths in Raqqa all over 2017 but increased in October and November, but not sudden peaks). Once we

Event Date	VDC Peak	Peak Type	War Event
Jul'15	ISIS	actor	Offensives against ISIS
Jul'16	ISIS	actor	Offensives against ISIS
Mar.'16 - end of 2016	Russian	actor	Russian Attack
May'16 - end of 2016	Syrian government	actor	Multiple Offensives
Feb.'15	warplane shelling	cause of death	Offensives against Kurds and ISIS
Jul'15	shooting	cause of death	Aleppo Offensive
Jul'16	shelling	cause of death	Aleppo Offensive
Mar.'14 - end of 2014	shooting	cause of death	Multiple Offensives
Aug.'13	chemical and toxic gases	cause of death	Ghouta Chemical Attack
Aug.'16	chemical and toxic gases	cause of death	Aleppo Chemical Attack
Oct.'17, Nov.'17	shooting	cause of death	The Raqqa Campain
Apr.'17	chemical and toxic gases	cause of death	Khan Sheikhoun Chemical Attack
Jul.'15	Aleppo	location	Aleppo Offensive
Apr.'17	Idlib	location	Khan Sheikhoun Chemical Attack
Jul.'16 - Aug.'16	Aleppo	location	Aleppo Offensive
Aug.'13	Aleppo	location	Ghouta Chemical Attack

Table 3.1: Major Events in the Syrian War Extracted From VDC

extracted these peaks, we researched the events that happened in Syria in the locations and dates of these peaks to find out the event that happened during that time. (E.g. the peak in chemical weapons in Aleppo marks the Aleppo chemical attack, and the peaks in Raqqa mark the Raqqa Campaign). Based on these observations from the VDC, we were able to extract some of the major events in the Syrian war as shown in table 3.1.

Finally, we scrape the sources from section 3.8.1 for the events as described in table 3.1. We use keywords relevant and specific to each of the events in order to make sure that we scrape all the articles reporting about this event.

Chapter 4

Labeling The Dataset

After scraping our sources for news about the Syrian war, we labeled our articles as fake or true in order to build a labeled dataset for training and testing. We achieved this by extracting war-related facts from each article in our dataset and compared these facts against a ground-truth source: VDC.

4.1 VDC

The VDC is a non-profit, non-governmental organization registered in Switzerland that tracks and documents human rights violations from the Syrian war[1]. *The VDC accepts funding solely from independent sources.* Since its onset in 2011, the VDC data records, in real time, war-related deaths as well as missing and detained people. As stipulated on its website, the VDC *adheres to international standards for the documentation of its data.*

The VDC relies on reports from investigators and a ground network of internationally trained field reporters, who attempt to cover every governorate in Syria. Reporters collect data in three steps. First, initial information on one or more victims is gathered, from immediate and local sources (for example, hospitals, morgues, accounts of relatives/friends, etc.). Second, supporting information such as videos or photographs are sought. With this, the account gets confirmed and a record gets established. The last step consists in actively investigating key information originally missing around the reported violation. For each death, the record consists of information relating to the demographics, date, location, cause of death (e.g., type of weapon used), and status of the victim (civilian or non-civilian). The latter status corresponds to any combatant, be that a member of the government forces, opposition forces, or other armed factions. Data is available in both Arabic and English, despite that inconsistencies may occur between the two databases.

[1]https://vdc-sy.net/en/

The VDC remains the *only human rights group documenting deaths in the Syrian conflict over the entire duration of the conflict*, and making the distinction between civilian or combatant status. It is also the *only one that endorses high risks in documenting the violations*. The VDC has been a source of valuable information for a wealth of notable public health publications on the human cost of the war in Syria (see [43–45] for a few examples).

Each record in the VDC database consists of the following fields:

- Name of causality
- Cause of death (e.g., shooting, shelling, chemical weapons, etc.)
- Gender and age group (i.e., adult male, adult female, child male, or child female)
- Type (civilian or non-civilian)
- Actor (e.g., rebel groups, Russian forces, ISIS, Syrian government and allies, etc.)
- Place of death (e.g. Damascus suburbs, Hama, Aleppo, etc.)
- Date of death

4.2 Extracting Article Claims

Now that we have a dataset of articles reporting about the Syrian war, we wish to extract facts from these articles in order to decide if their claims are credible or fake. Since we are working with news related to armed conflict, we extracted the following war-related facts from each of the articles in our dataset:

1. date of the event reported in the article
2. location of the event as reported in the article
3. number of civilian casualties
4. number of children casualties
5. number of women casualties
6. number of noncivilian casualties
7. cause of death
8. actor (blame for the casualties)

4.2.1 Figure-Eight Job

In order to extract the war-related facts that we are interested in from each article in our dataset, we launched a crowdsourcing job using the crowdsourcing platform Figure Eight [2]. In particular, for each news article in our dataset, we ask three contributors on Figure Eight to answer the following questions:

1. When does the article claim the deaths happened? (day, month, and year)
2. Where does the article claim the deaths happened? (location). Answer can be one of the provinces in Syria:
   - Aleppo
   - Damascus
   - Damascus Suburbs
   - Daraa
   - Deir Ezzor
   - Hama
   - Hasakeh
   - Idlib
   - Lattakia
   - Quneitra
   - Raqqa
   - Sweida
   - Tartous
   - Article does not specify (in case the article does not specify the province where the event happened)
3. How many civilians died in the incident?
4. How many children died in the incident?
5. How many adult women died in the incident?
6. How many non-civilians died in the incident?
7. How did the casualties die? (cause of death). Answer can be one of the following causes of death:
   - Chemical and toxic gases

[2]https://www.figure-eight.com/

- Execution
- Explosion
- Shelling
- Shooting
- Siege
- Warplane shelling
- other (in case the cause of death specified in the article is none of the above)
- Article does not specify (in case the cause of death of the casualties is not specified in the article)

8. Who does the article blame for the casualties? (actor). Answer can be one of the following actors:
    - Al-Nusra Front
    - Armed opposition groups
    - Russian troops
    - Self administration forces (Kurdish Forces)
    - Syrian government and affiliated militias
    - The organization of the Islamic State in Iraq and the Levant - ISIS
    - International coalition forces (also known as the US-led coalition forces)
    - Article does not specify (in case the article does not specify or claims the actor is unknown)

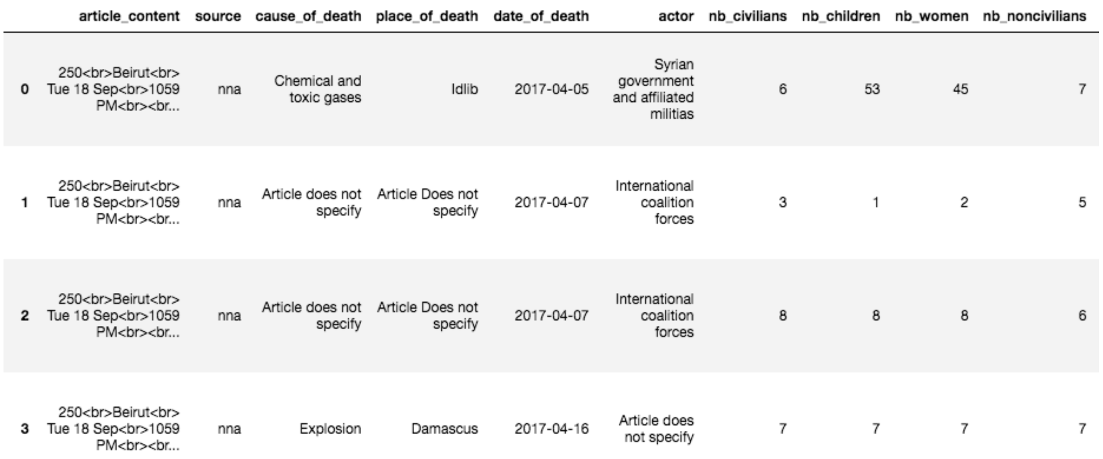
Aggregating The Answers

Once the crowdsouring task was completed, we aggregate the answers given to us by the three crowd workers for each article. That is, for each question asked about each article, we take the majority vote for the answer as our final **aggregated crowd answer**.

Quality Control

In order to ensure the quality of the answers we receive from the crowdsourcing workers, we required each article in the dataset to be answered by three contributors. We also required these contributors to be of the Level 3 contributors, who are a small group of contributors on Figure Eight with the most experience and the highest accuracy on past contributions [3].

[3]https://www.figure-eight.com/



	article_content	source	cause_of_death	place_of_death	date_of_death	actor	nb_civilians	nb_children	nb_women	nb_noncivilians
0	250 Beirut Tue 18 Sep 1059 PM <br...	nna	Chemical and toxic gases	Idlib	2017-04-05	Syrian government and affiliated militias	6	53	45	7
1	250 Beirut Tue 18 Sep 1059 PM <br...	nna	Article does not specify	Article Does not specify	2017-04-07	International coalition forces	3	1	2	5
2	250 Beirut Tue 18 Sep 1059 PM <br...	nna	Article does not specify	Article Does not specify	2017-04-07	International coalition forces	8	8	8	6
3	250 Beirut Tue 18 Sep 1059 PM <br...	nna	Explosion	Damascus	2017-04-16	Article does not specify	7	7	7	7

Figure 4.1: Screenshot of the Crowd Answers

In addition, we made use of Figure Eight's test questions and quiz mode features. The quiz mode allows us to provide the contributors with a quiz that they have to pass in order to participate in our job. On the other hand, the test questions allow us to assign to each page of five articles one test question that we have previously annotated with *gold* answers. These test questions allow us to ensure that the contributors are annotating the articles properly. The answers provided by the contributors for these test questions are compared against our gold answers. We required the contributors to have an accuracy above 70%. Contributors who have an accuracy less than the threshold are automatically removed from the job.

In order to verify the quality of the job performance, we calculate the Fleiss Kappa agreement between the answers provided by the crowdsourcing contributors for each of the questions asked. Table 4.1 shows that the agreement of our contributors ranges from moderate to perfect on all of the questions in our job.

Finally we export the results of the crowd answers as an aggregated report, which consists of the aggregated responses for each question for each individual row in the job. This allows us to extract the majority vote in each answer to each question for each article. Figure 4.1 shows what our dataset looks like once the crowdsourcing job completes.

As we will see in the coming sections, our labeling mechanism depends on how accurate an article is when reporting casualties in a certain war-related event. For this reason, we drop all articles in our dataset that do not report casualties. In order to achieve this, once the crowdsourcing task is complete, we drop the articles where all the numbers of casualties (numbers of women, children, civilians, and non civilians) are all zeros or the cause of death is no specified, i.e. the article does not report deaths.

Table 4.1: Fleiss Kappa Agreement of the Contributors for Each Question

Question	Fleiss' Kappa Measure
Number of Civilian Casualties	0.67
Number of Children Casualties	0.50
Number of Women Casualties	0.75
Number of Non-Civilian Casualties	0.56
Cause of Death	0.66
Actor	0.74
Place of Death Claim	0.51
Day	0.92
Month	1
Year	1

Stats About Crowdsourcing Answers

In this section, we take a look at the percentage of articles in our dataset where all three workers agreed on the answer to a question, two out of the three workers agreed on the answer, and all three workers disagreed on the answer for each of the questions we asked.

1. Number of Civilian Casualties:

   - 70% of the articles in our dataset had full agreement (all 3 workers agreed on the answer)
   - 20% of the articles in our dataset had partial agreement (2 out of 3 workers agreed on the answer)
   - 10% of the articles in our dataset had no agreement (all 3 workers disagreed on the answer)

2. Number of Children Casualties:

   - 80% of the articles in our dataset had full agreement (all 3 workers agreed on the answer)
   - 10% of the articles in our dataset had partial agreement (2 out of 3 workers agreed on the answer)
   - 10% of the articles in our dataset had no agreement (all 3 workers disagreed on the answer)

3. Number of Women Casualties:

- 80% of the articles in our dataset had full agreement (all 3 workers agreed on the answer)
- 10% of the articles in our dataset had partial agreement (2 out of 3 workers agreed on the answer)
- 10% of the articles in our dataset had no agreement (all 3 workers disagreed on the answer)

4. Number of Children Casualties:

   - 70% of the articles in our dataset had full agreement (all 3 workers agreed on the answer)
   - 20% of the articles in our dataset had partial agreement (2 out of 3 workers agreed on the answer)
   - 10% of the articles in our dataset had no agreement (all 3 workers disagreed on the answer)

4.2.2 Manual Quality Control

Once the crowdsourcing task was completed, we went over the articles and answered the questions that we asked the crowdsourcing workers in order to compare their answers to the gold answers and evaluate the quality of our results. The following were the mistakes made by the crowdsourcing workers for each of our questions.

- 40% of the workers provided a wrong number for civilian casualties.
- 66% of the workers provided a wrong number for children casualties.
- 70% of the workers provided a wrong number for women casualties.
- 50% of the workers provided a wrong answer for cause of death.
- 50% of the workers provided a wrong answer for actor.
- 33% of the workers answered “article does not specify” for actor even though the actor was specified in the article.
- 34% of the workers answered “article does not specify” for cause of death even though the cause of death was specified in the article.
- 10% of the workers randomly selected or assumed the actor even though the article does not specify the actor.

- 10% of the workers randomly selected or assumed the cause of death even though the article does not specify the cause of death.

However, the above list of mistakes led to mistakes in our aggregated crowd answers. After comparing the aggregated answers to the gold answers we manually extracted for each article, we realized the following mistakes in our aggregated crowd sourcing answers.

- 50% of the aggregated crowd answer for number of civilian casualties were wrong.
- 23% of the aggregated crowd answer for number of children were wrong.
- 14% of the aggregated crowd answer for number of women were wrong.
- 30% of the aggregated crowd answer for number of non-civilians were wrong.
- 50% of the aggregated crowd answer for cause of death were wrong.
- 50% of the aggregated crowd answer for actor were wrong.

In conclusion, the main mistakes made by the crowd workers were the following:

- The majority of the numerical answers were correct, with the exception of few workers writing random numbers. Numbers were probably easier to extract from the articles since most articles had the numbers either in the title or in the very first lines. E.g. An article titled “10 civilians dead in latest attack on Aleppo” makes it easier, quicker, and clearer to extract the number of civilians from this article.
- Choosing “Article does not specify” option from the drop-down menu in articles where the cause of death or actor was specified somewhere in the middle of the text. This shows that the workers were probably only reading the first few lines of an article and answering based on them.
- Workers were not reading the complete articles to properly find the answers to our questions. This can be avoided by selecting shorter articles of one or two paragraphs to make sure the readers read the entire article.
- Randomly selecting answers from drop-down menus. To increase the quality of our answers in the future, we can use strict test questions for each article in our dataset. An example can be to make the worker select from a drop-down menu the big scale event that the article is reporting about. Answering such a question requires the worker to read the entire text to select the correct answer. This allows us to drop the workers who randomly select answers without reading the articles.

By reading the entire dataset of articles and manually extracting the answers to our questions ourselves, we made sure that the above mistakes did not affect the quality of our answers or our results. For the next sections, we use our gold answers in all of our experiments.

4.3 Mapping Article Claims to the VDC Casualties

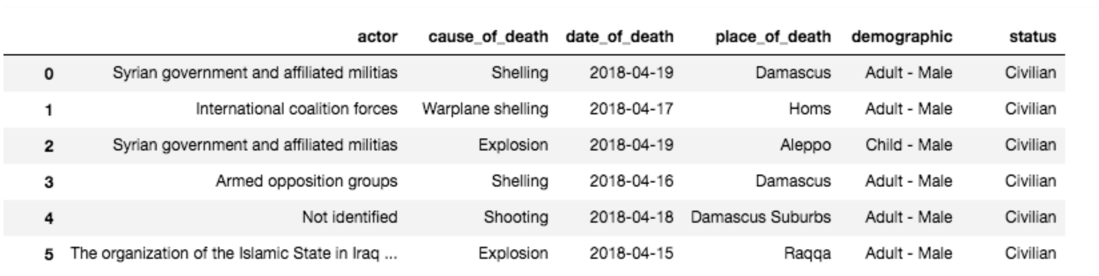
4.3.1 Aggregating the VDC Casualties into Events

As discussed in section 4.1, the VDC consists of a database reporting the details about the casualties in the Syrian war. It consists of a table showing the actor (blame for the casualty), cause of death, date of death, place of death, demographic (adult male, adult female, child male, or child female), and status (civilian or noncivilian) of each person who died in the Syrian war. Figure 4.2 shows a screenshot of this table.

On the other hand, as described in section 4.2, we extracted from each article claim the number of civilian, non-civilian, children and women casualties along with the actor, cause of death, and the date and place of the event that the article is reporting. For this reason, in order to properly map the article to the VDC event it is reporting, we created war events from the VDC casualties that have the required structure to allow us to perform the mapping properly. We achieved this by aggregating the casualties in the VDC and grouping them by actor, cause of death, date of death, and place of death, while counting the numbers of civilians, non-civilians, children, and women. The following SQL statement performs this aggregation:

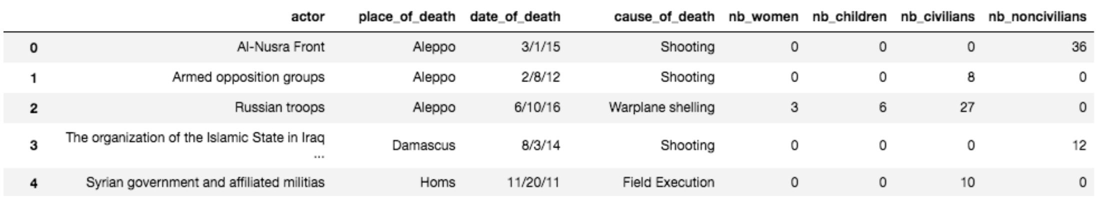
"SELECT actor, place_of_death, date_of_death, cause_of_death,
COUNT(CASE demographic WHEN 'Adult Female' THEN 1 ELSE null END) AS nb_women,
COUNT(CASE demographic WHEN 'Child Male' THEN 1 WHEN 'Child Female' THEN 1 ELSE null END) AS nb_children,
COUNT(CASE status WHEN 'Civilian' then 1 ELSE null END) AS nb_civilians,
COUNT(CASE status WHEN 'NonCivilian' THEN 1 ELSE null END) AS nb_noncivilians
FROM vdc_data GROUP BY actor, place_of_death, date_of_death".

We run this SQL statement using pandasql's pysql on the dataframe of the VDC casualties vdc_data. Now our VDC casualties are grouped into events as show in figure 4.3 and thus have the same structure as our crowd sourcing answers.



	actor	cause_of_death	date_of_death	place_of_death	demographic	status
0	Syrian government and affiliated militias	Shelling	2018-04-19	Damascus	Adult - Male	Civilian
1	International coalition forces	Warplane shelling	2018-04-17	Homs	Adult - Male	Civilian
2	Syrian government and affiliated militias	Explosion	2018-04-19	Aleppo	Child - Male	Civilian
3	Armed opposition groups	Shelling	2018-04-16	Damascus	Adult - Male	Civilian
4	Not identified	Shooting	2018-04-18	Damascus Suburbs	Adult - Male	Civilian
5	The organization of the Islamic State in Iraq ...	Explosion	2018-04-15	Raqqa	Adult - Male	Civilian

Figure 4.2: Screenshot of the Data Extracted from VDC



	actor	place_of_death	date_of_death	cause_of_death	nb_women	nb_children	nb_civilians	nb_noncivilians
0	Al-Nusra Front	Aleppo	3/1/15	Shooting	0	0	0	36
1	Armed opposition groups	Aleppo	2/8/12	Shooting	0	0	8	0
2	Russian troops	Aleppo	6/10/16	Warplane shelling	3	6	27	0
3	The organization of the Islamic State in Iraq ...	Damascus	8/3/14	Shooting	0	0	0	12
4	Syrian government and affiliated militias	Homs	11/20/11	Field Execution	0	0	10	0

Figure 4.3: Screenshot of the VDC Events After Aggregation

4.3.2 Mapping Each Article to a VDC Event

Now that we have the VDC events and the article claims in the same structure, we can map each article to the event it is reporting. In order to achieve this, for each article, we need to find the VDC event closest to this article's claim.

We first started by mapping the article claim to all the VDC events that take place in the same event within a day range window. For example, if according to the article claim the place of death is Idlib and the date of death is 04/04/2017, we map the article to all the events found in the VDC that are within *day_range* days from the date event 04/04/2017 such that they took place in the claimed place of death which is Idlib. In order to achieve this, we run the following SQL statement using pandasql's pysql:

```
"SELECT * FROM vdc_events
WHERE vdc_events.place_of_death = 'place_of_death'
AND vdc_events.date_of_death >= 'start_date'
AND vdc_events.date_of_death <= 'event_date"
```

Where *place_of_death* is the place of death in the article claim, *start_day* is *day_range* days before the date in the article claim, and *event_day* is the date in the article claim. We decide the correct *day_range* by experiment.

Once we run this SQL statement, we end up with a number of events mapped to the article claim. In case the article is not mapped to any event, we drop this article from our dataset. Both of the article claim and the VDC events have the following structure:

[nb_civilians, nb_children, nb_women, nb_noncivilian, actor, cause_of_death]

For this reason, in order to find the VDC event closest to the article claim out of all of the mapped events, we calculate the Gower distance between the article claim and each of these events, and choose the event with the least Gower distance. We chose to use Gower distance because we are dealing with both numerical (nb_civilians, nb_children, nb_women, nb_noncivilians) and categorical (actor, cause_of_death) features.

4.4 Clustering The Articles

Once we mapped each article with its closest VDC event, we labeled each article as fake or credible based on the event it was mapped with. In order to achieve this, we clustered the articles into two groups: fake and credible.

4.4.1 Setting The Clustering Features

We first started by setting the article's clustering features based on the event it was mapped with. As discussed earlier, each of the article claims and the VDC events have the following structure:

[nb_civilians, nb_children, nb_women, nb_noncivilian, actor, cause_of_death]

For this reason, we set the clustering features of each article to be the difference between the article claim and the VDC event it was mapped with for each of the values [nb_civilians, nb_children, nb_women, nb_noncivilian, actor, cause_of_death].

For the numerical features (nb_civilian, nb_women, nb_children, nb_noncivilian) we set the feature to be the absolute value of the difference between the number in the article claim and the number in the event, divided by the number of the event. For example, the nb_civilians feature is equal to abs(article_claim[“nb_civilian”] - vdc_event[“nb_civilians”]) / vdc_event[“nb_civilians”]. Dividing the difference by the actual number allows us to set the feature to describe how accurately an article reports the casualties in the event.

As for the categorical values (actor and cause of death), since we are dealing with distances, we set the value of the feature to be equal to 0 if the article claim is the same as that in the event, and 1 otherwise.

Since we are clustering elements that have both categorical and numerical features, we perform minmax normalization on the articles' numerical features. Once the feature calculation and the normalization is complete, we are now ready to cluster the articles into fake and credible.

4.5 Experiments and Results

As discussed in section 4.3.2, we experiment with different *day_range* window sizes to find the window size that gives us the best clustering results. We tested out window sizes from *day_range = 1* to *day_range = 10* and found that the best window size of the highest silhouette coefficient of 0.67 is *day_range = 4*. Figure 4.4 shows the silhouette coefficient of the clusters for each window size. Figure 4.5 shows the PCA plot of the two clusters.
Out of 804 articles in our dataset, 378 articles belonged to cluster 1, and 426 articles belonged to cluster 2.

Now that our articles are split into two clusters, we need to decide which of the clusters corresponds to the credible articles, and which one corresponds to the fake articles. We decide this based on the cluster centers. Since our clustering features were the distances between the article claims and the VDC event it is reporting, this means that the center with the lower features is for the credible articles cluster (closer to the truth), and the center with the higher features is for the fake article cluster (further from the truth). The following are the feature values of each of the centers of our clusters:

Cluster 1: [0.03, 0.01, 0.01, 0.02, 0.5, 1]

Cluster 2: [0.001, 0.002, 0.001, 0.001, 0.03, 0]

Where each cluster center is in the form: [nb_civilians, nb_children, nb_women, nb_noncivilian, actor, cause_of_death]

The above cluster centers allow us to notice the following: The articles in cluster 1 have a higher distance in number of civilian casualties than the articles in cluster 2 (0.03 vs 0.001), a higher distance in number of children casualties (0.01 vs 0.002), a higher distance in number of women casualties (0.02 vs 0.001). Recall that the actor and cause of death feature for each article are equal to 0 if the article and the VDC claim are equal and equal to 1 if the article and the VDC claim are different. The centers also show that the articles in cluster 1 are more likely to have a wrong actor than the articles in cluster 2 (actor closer to 1 in center 1 whereas actor closer to 0 in cluster 2). In addition, the centers show that the articles in cluster 1 all have wrong cause of death, whereas the articles in cluster 2 all have correct cause of death. Thus we conclude that the cluster 2 is the credible articles cluster and cluster 1 is the fake articles cluster.

Therefore, we have now built a dataset of 804 articles, where 378 were labeled **fake** and 426 were labeled **true**. The plot of figure 4.6 shows for each of the clustering features, the number of articles labeled fake that had this feature different from the event matched with VDC.

4.5.1 Examples

An Article Labeled Fake In Our Output

03-08-2016 Chemical Attack Kills Five Syrians in Aleppo. At least five Syrians have been killed and a number of others injured in a chemical attack by foreign-sponsored Takfiri militants against a residential neighborhood in northwestern Syria. At least five Syrians have been killed and a number of others injured in a chemical attack by foreign-sponsored Takfiri militants against a residential neighborhood in northwestern Syria. Health director for Aleppo Mohammad Hazouri said five people died and eight others experienced breathing difficulties after artillery shells containing toxic gasses slammed into the Old City of Aleppo on Tuesday the official SANA news agency reported. Government sources said Takfiri terrorists had also used chemical munitions against civilians in the city of Saraqib in the Idlib province but militants accused government forces of carrying out the attack. Doctor Ibrahim al-Assad a neurologist in Saraqib said he treated 16 of 29 cases brought to his hospital on Monday night. He added that most of the victims were women and children and were suffering from breathing difficulties red eyes and wheezing. Rescuers and doctors in the city said the symptoms were similar to those caused by chlorine gas. The chemical raids come as the Syrian army is making progress in operations to retake Aleppo from militants who are seeing the noose tightening around them in the areas which they control.

- Source: Al Manar
- location based on article: Aleppo
- Actor based on article: unknown (claims terrorist organization but does not name the organization)
- Cause of death based on article: chemical and toxic gases
- Number of civilian casualties based on article: 5
- Numbers of children, women and non-civilians: 0

When clustering was performed, this article was mapped with the following VDC event:

- location: Aleppo

- number of civilians: 6
- number of children, women, non-civilians: 0
- actor: unknown
- cause of death: shooting

As we can see, the closest event in VDC to this article states that it is true that around 5 civilians were killed by an unknown organization in Aleppo. However, these civilians were killed by shooting and not by chemical and toxic gases. For this reason, this article was labeled fake by our method.

An Article Labeled True In Our Output

July 19 2016 A Syria Democratic Forces (SDF) fighter walks in the silos and mills of Manbij after the SDF took control of it in Aleppo Governorate Syria July 1 2016. Airstrikes on Daesh-held villages in northern Syria killed at least 85 civilians on Tuesday as intense fighting was underway between the militants and U.S-backed fighters Syrian opposition activists and the extremist group said. Residents in the area blamed the U.S.-led coalition for the strikes that targeted two villages Tokhar and Hoshariyeh which are controlled by IS activists said. The villages are near the Daesh stronghold of Manbij a town that members of the PYD-dominated U.S.-backed Syria Democratic Forces (SDF) have been trying to capture in a weeks-long offensive. The Britain-based Syrian Observatory for Human Rights said at least 56 civilians including 11 children were killed in the strikes on the villages which also wounded dozens. Another activist group the Local Coordination Committees said dozens of civilians mostly families were killed. Turkeys official Anadolu Agency put the death toll at least at 85 adding that 50 civilians were also wounded in airstrikes. The Daesh-linked Aamaq news agency claimed 160 civilians mostly women and children were killed in Tokhar alone in a series of purportedly American airstrikes around dawn Tuesday. Postings on a Facebook page show images of people including children as they were being put in collective grave purportedly in the village of Tokhar. One photograph shows a man carrying the lifeless body of a child covered with dust while another shows a child partly covered by a blanket lying in a grave. Tuesdays casualties come on the heels of similar airstrikes on the Daesh-held town of Manbij on Monday when at least 15 civilians were reportedly killed. Meanwhile the headquarters of Daesh militants inside Manbij was captured as SDF forces pushed into the western part of the town over the weekend the U.S. military said in a statement on Tuesday. The headquarters which was located in a hospital was being used as a command center and logistics hub. The U.S.-backed Syrian rebels also took control of part of the town enabling civilians in the area to flee the fighting the statement said. The rebels were continuing to battle Daesh on four fronts for control of Manbij clearing territory as they pushed toward the center of the city the statement said. Daesh militants have staged counterattacks but the Syrian rebels have maintained momentum with the help of air strikes by the U.S.-led coalition the statement said. It said the coalition has carried out more than 450 air strikes around Manbij since the operation to take the town began. The U.S. Central Command said the coalition conducted 18 strikes on Monday and destroyed 13 Daesh fighting positions seven Daesh vehicles and two car bombs near Manbij. The Manbij area has seen intense battles between Daesh extremists and the Kurdish-led fighters who have been advancing under the cover of intense airstrikes by the U.S.-led coalition. If Manbij is captured by the U.S.-backed fighters it will be the biggest strategic defeat for Daesh in Syria since July 2015 when the extremist group lost the border town of Tal Abyad. In neighboring Iraq meanwhile Daesh has been beaten back on several fronts with Iraqi forces aided by U.S.-led coalition airstrikes having retaken the cities of Ramadi and Fallujah in western Anbar province. Coalition airstrikes kill 85 civilians in Daesh-held villages in Syrias Manbij

- Source: Daily Sabah
- location based on article: Aleppo
- Actor based on article: international coalition forces
- Cause of death based on article: warplane shelling
- Number of civilian casualties based on article: 85
- Number of children casualties based on article: 11
- Numbers of women and non-civilians: 0

When clustering was performed, this article was mapped with the following VDC event:

- location: Aleppo
- number of civilians: 69
- number of children: 9
- number of women and non-civilians: 0
- actor: international coalition forces
- cause of death: warplane shelling

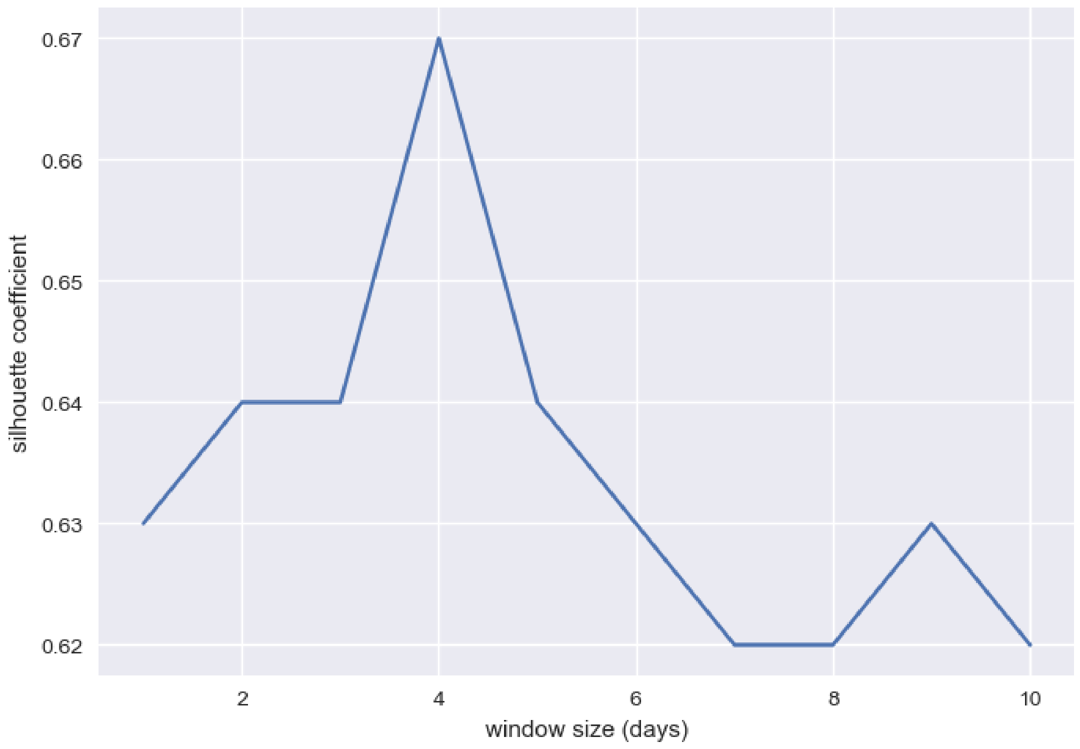


Figure 4.4: Figure showing the silhouette coefficient of the clusters for each window size

As we can see, the closest event in VDC to this article states that it is true that around 80 civilians were killed by the international coalition's warplane shelling in Aleppo. The article has the correct actor, cause of death, an almost correct number of children and a very close number of civilians. Therefore, this article was labeled true by our method.

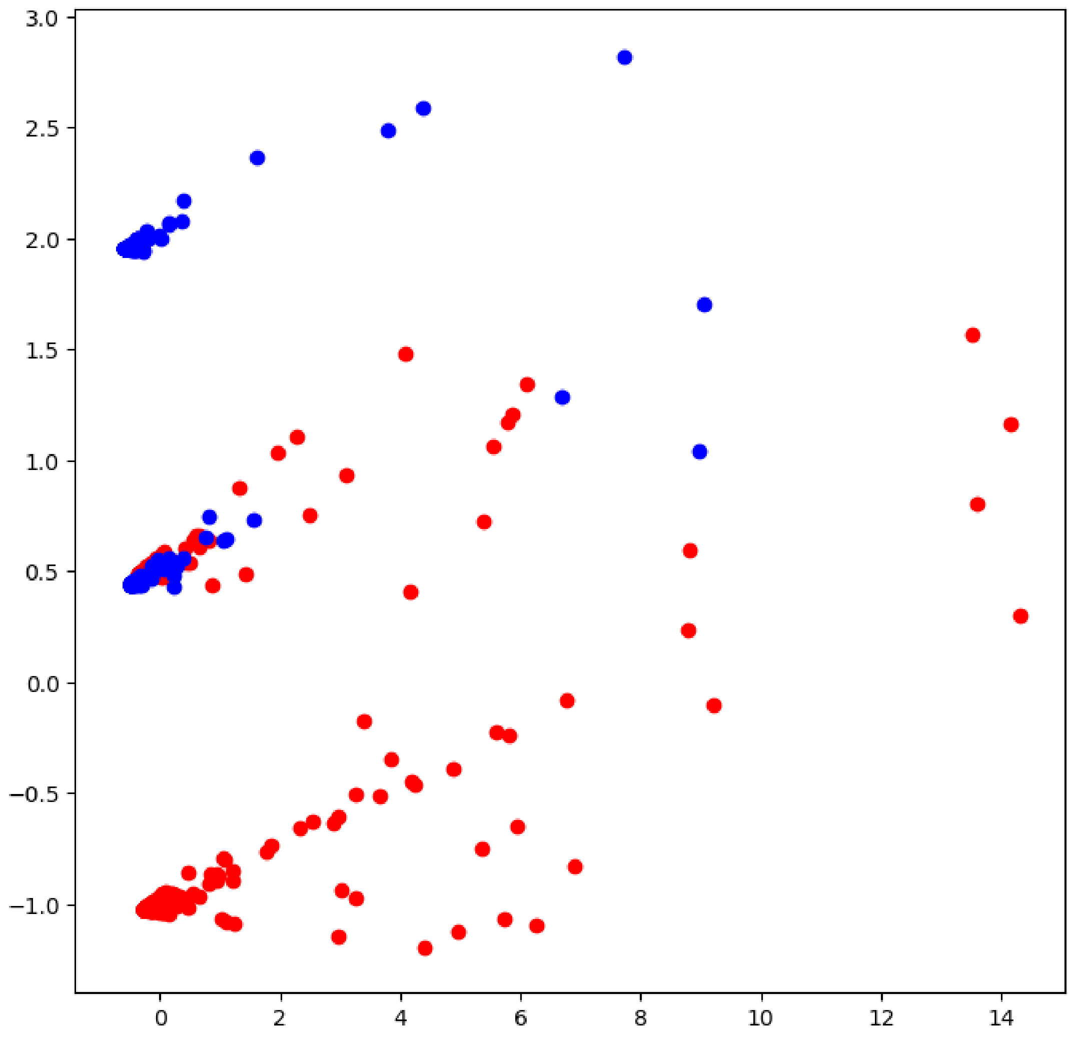


Figure 4.5: Figure showing the PCA plot of the clusters

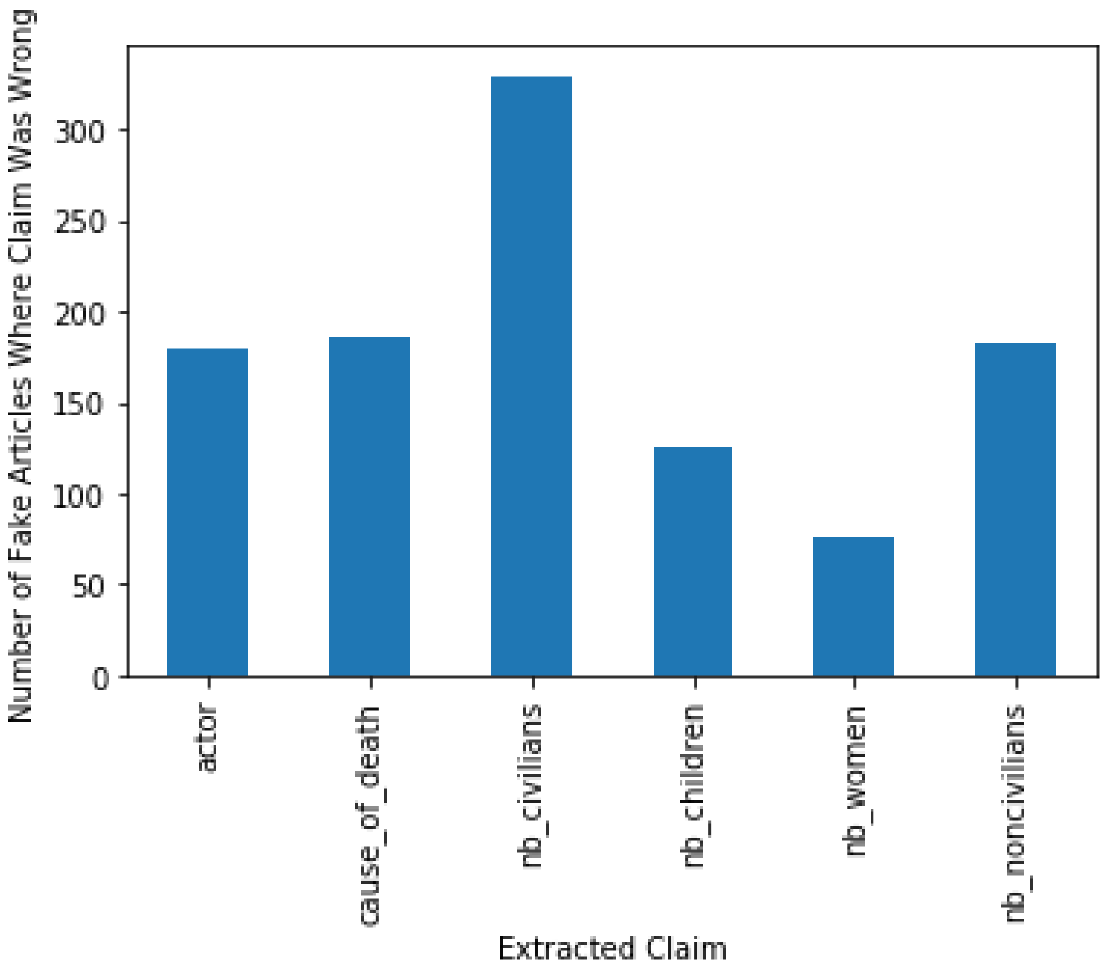


Figure 4.6: Plot Showing The Number of Fake Articles Where A Clustering Feature Was Different From VDC

Chapter 5

Feature Extraction

Once we built and labeled our dataset, we performed feature extraction on it in preparation to test and train a machine learning model. Once we extracted all the features we are interested in from our dataset, we split our dataset into 80% for training and 20% for testing. We then performed exploratory analysis on our dataset in order to visualize the distribution of the values of each of our features in the articles which were labeled true and the articles which were labeled fake. In this chapter, we describe the features of interest that we extracted from our dataset and report the results of exploratory analysis done on the dataset.

5.1 Stats About the Label

We first take a look at the labels of the articles in our training dataset. Figure 5.1 shows the number of articles labeled true and number of articles labeled fake in our training data independent of the source or category. In our dataset, 53% of our articles were labeled true, and 47% of articles were labeled fake.

5.1.1 Stats About the Events and Dates

Events scraped for were the following:

- E1: Ghouta Chemical Attack
- E2: 2014 Offensives
- E3: Offensives against Kurds and Offensives against ISIS
- E4: Major Offensives against ISIS in July 2015
- E5: Aleppo Offensive and Major Offensives against ISIS
- E6: Multiple Offensives All Over Syria/Russian Attack on Syria

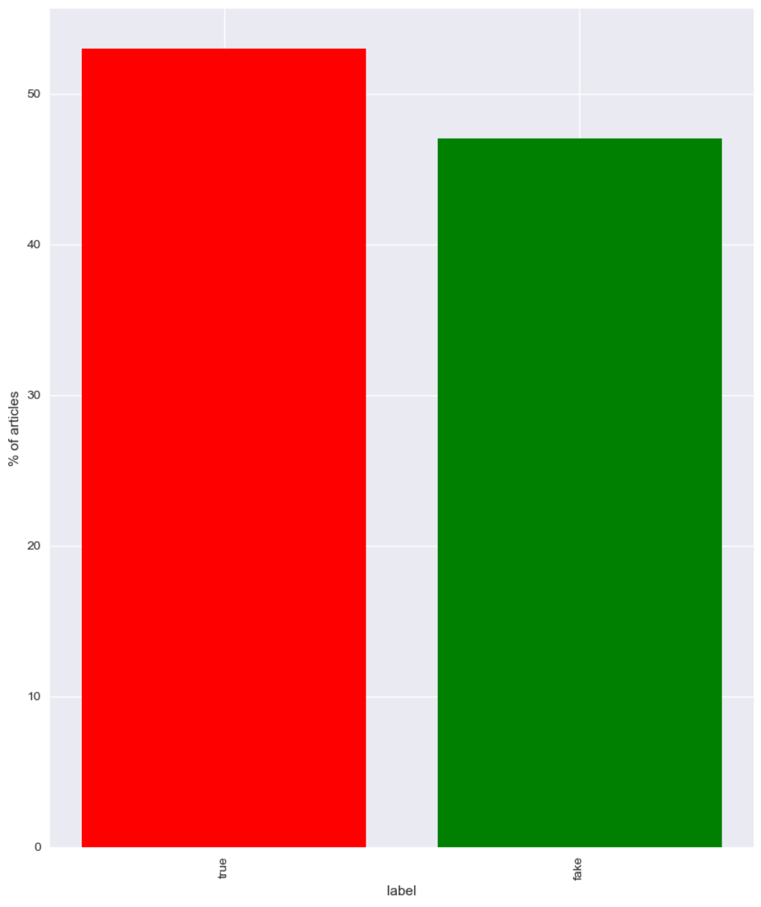


Figure 5.1: Plot Showing the number of articles labeled true and number of articles labeled fake independent of the source or category.

- E7: Aleppo Chemical Attack
- E8: Khan Sheikhoun Chemical Attack
- E9: The Raqqa Campain

Figure 5.4 shows the number of articles labeled fake per month during the Syrian war. From figure 5.4 and table 3.1, we notice that the dates with the peaks of fake articles from our dataset were during the following events:

- April 2017: the same time as Khan Sheikhoun Chemical Attack
- Summer 2016: the same time as Aleppo offensive and other major offensives against ISIS
- August and September 2016: the same time as the Aleppo chemical attack
- August and September 2013: the same time as the Ghouta chemical attack

On the other hand, we notice that the dates with the least number of fake articles from our dataset were the following events:

- Year 2014: at the same time as multiple offensives including Lattikiah, Qalamoun, etc..
- Year 2016: at the same as the early stages of the involvement of Russian forces in the Syrian war
- Summer 2015: at the same time as the offensives against ISIS
- Oct Nov 2017: at the same time as the Raqqa Campaign against ISIS

5.1.2 Stats About the Sources

As discussed in chapter 3, we made sure to include in our dataset all the articles that were reported about the events we were interested in from the Syrian war. However, as we discussed in chapter 4, we are only working with articles that are reporting the war events in which there were casualties, and therefore some articles had to be dropped from our dataset for not reporting casualties. A lot of the sources we have focus when reporting about the Syrian war about political statements, opinions, advancement of certain parties over the others in war, etc... which we do not deal with in this work. For this reason, some sources have less articles in our dataset than others. We kept these sources despite the fact that they did not provide us with as much articles as others in order to make sure we had in our dataset as much of diversity when it comes to the opinions about the Syrian war as possible.

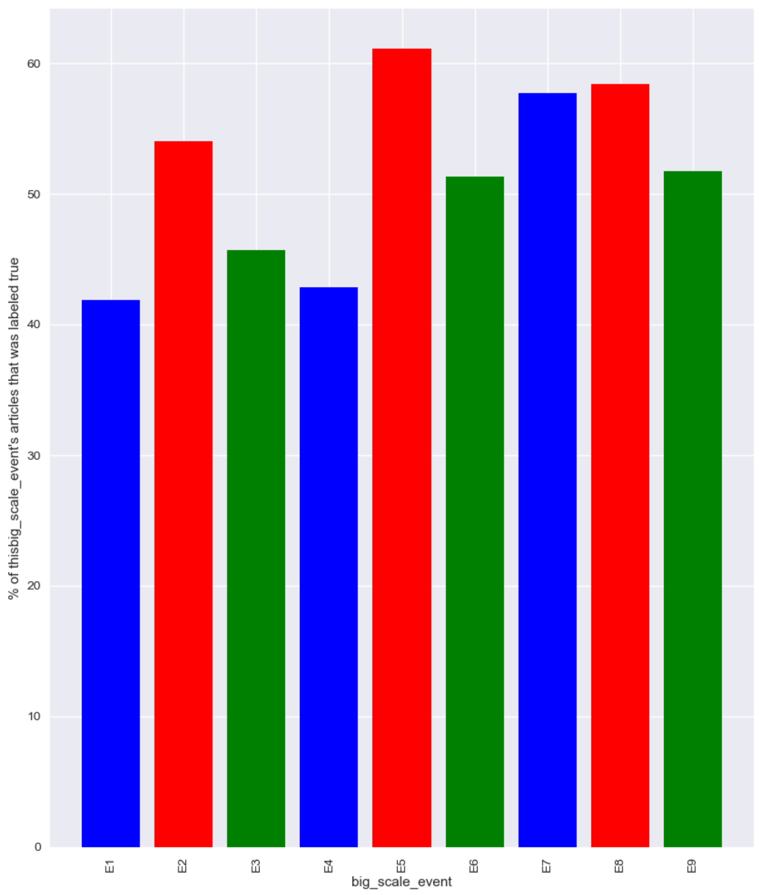


Figure 5.2: Plot Showing the % of Articles Labeled True for Each Event where % = nb_true_per_event / nb_total_per_event

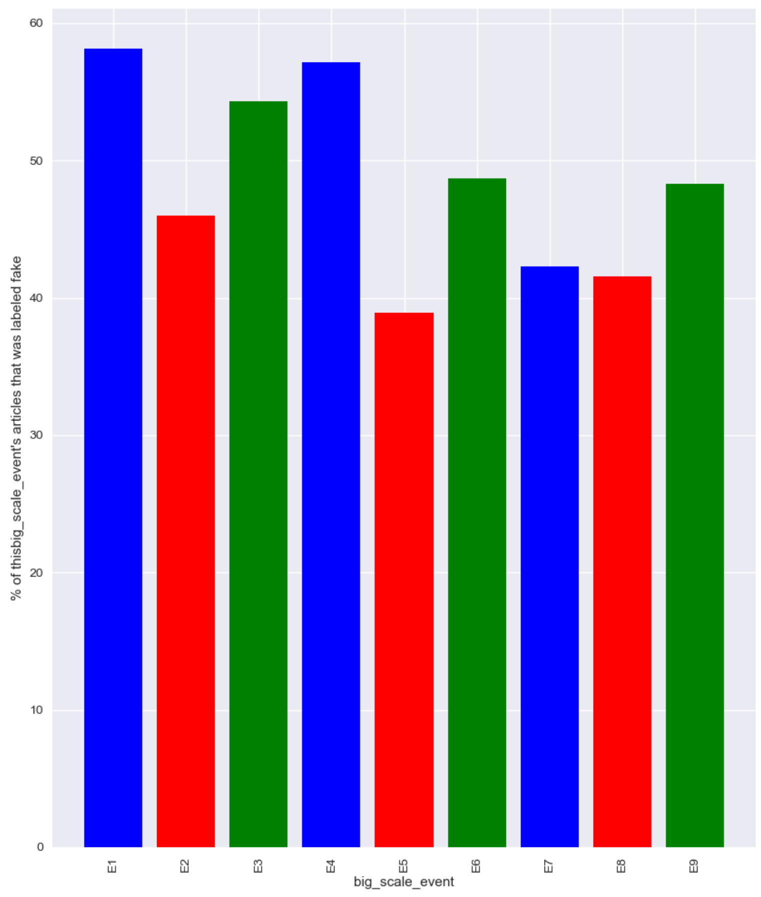


Figure 5.3: Plot Showing the % of Articles Labeled Fake for Each Event where % = nb_fake_per_event / nb_total_per_event

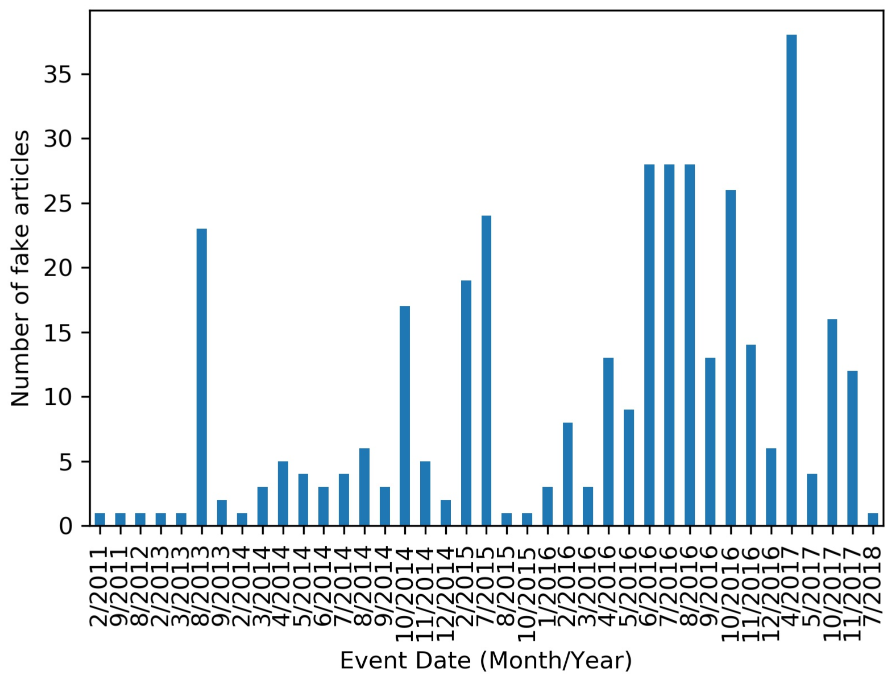


Figure 5.4: Plot Showing the Number of Articles Labeled Fake For Each Month

5.2 List of Features Extracted

The following list summarizes the features we extracted from our dataset. Some of these features, as we will see in the coming sections, were based on the related work done by [6] as described in literature review chapter 2. In the coming sections, we specify which features were based on the related work and which were added by us.

1. Sectarian language
2. Consistency Score
3. Description of sources quoted in reporting the news
4. Assertive verbs
5. Factive verbs
6. Hedges
7. Implicative verbs
8. Report verbs
9. Subjectivity and bias

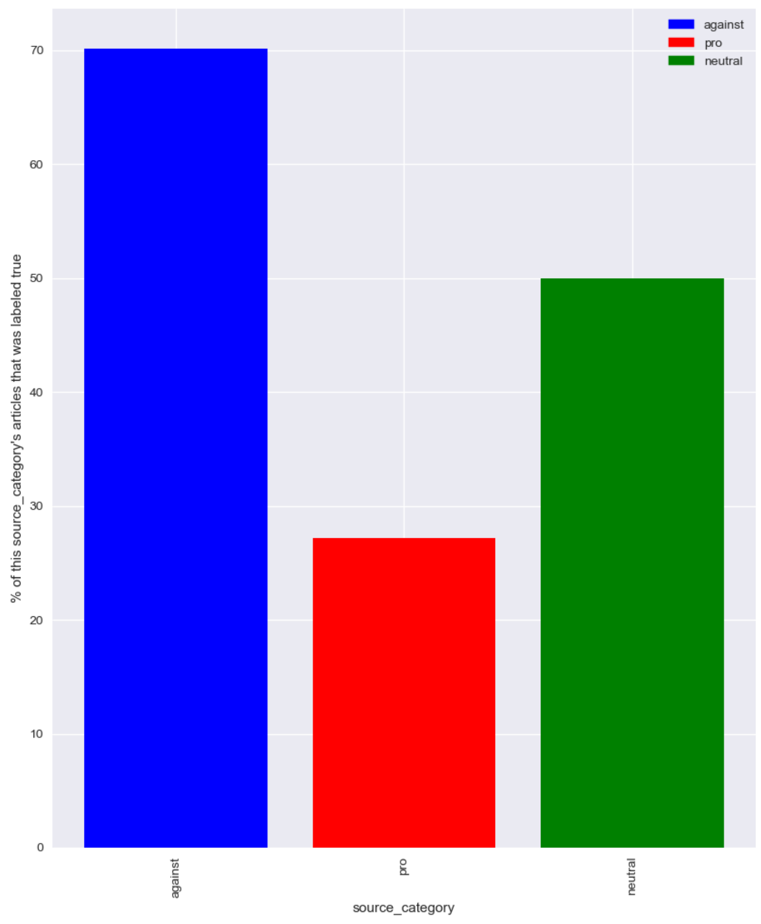


Figure 5.5: Plot Showing the % of Articles Labeled True for Each Category where % = nb_true_per_category / nb_total_per_category

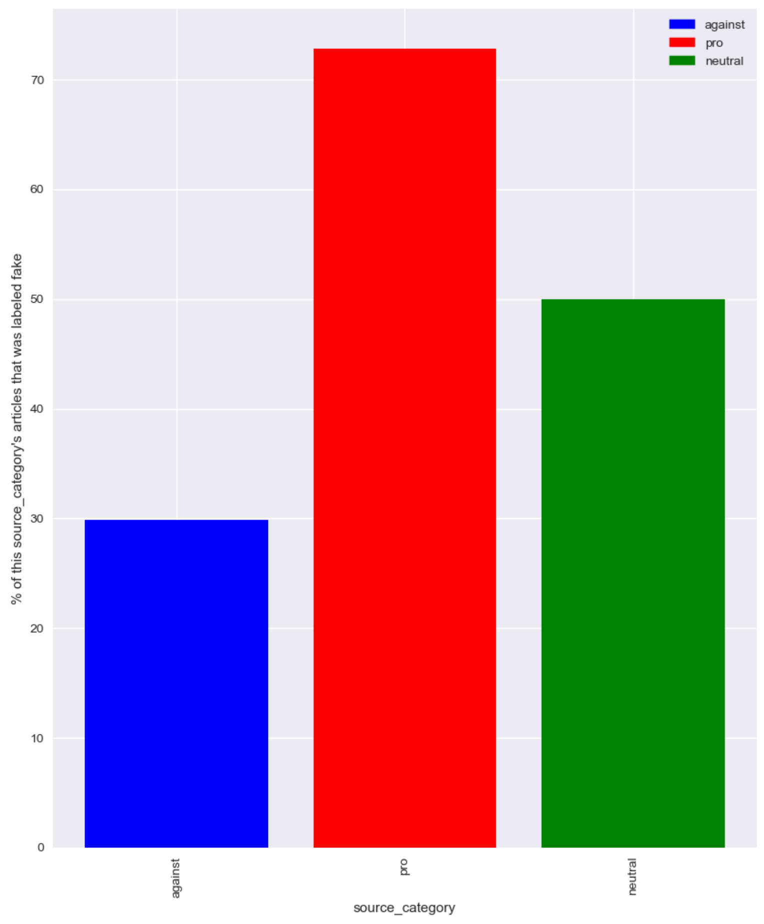


Figure 5.6: Plots Showing the % of Articles Labeled True/Fake for Each Source Category (Pro, Against, Neutral)

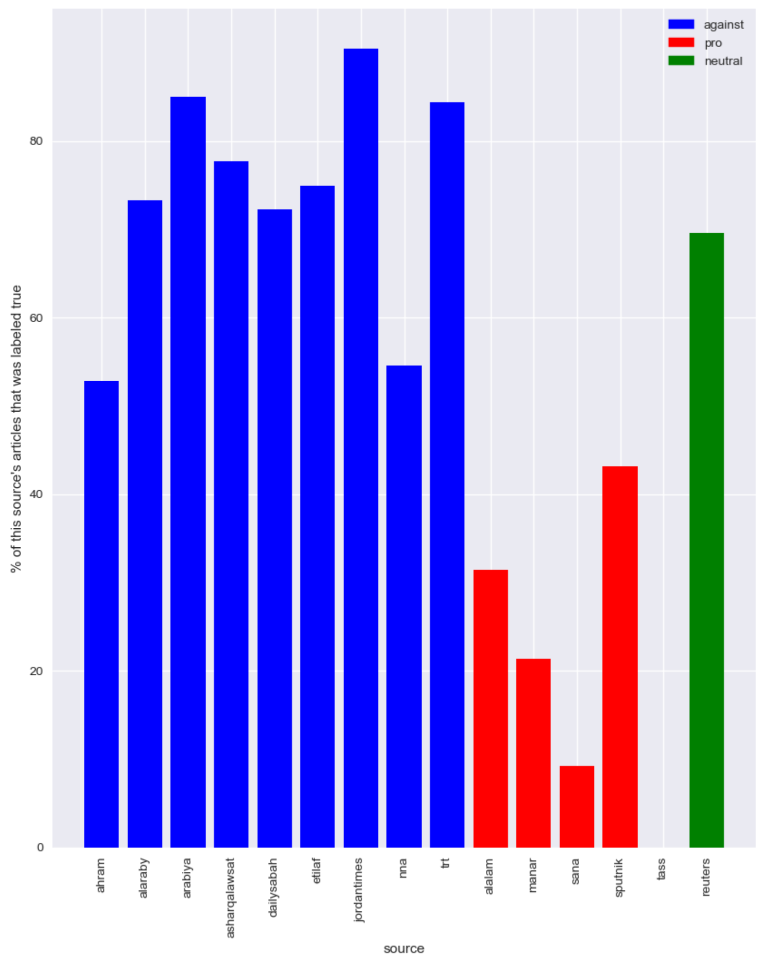


Figure 5.7: Plot Showing the % of Articles Labeled True for Each Source where % = nb_true_per_source / nb_total_per_source

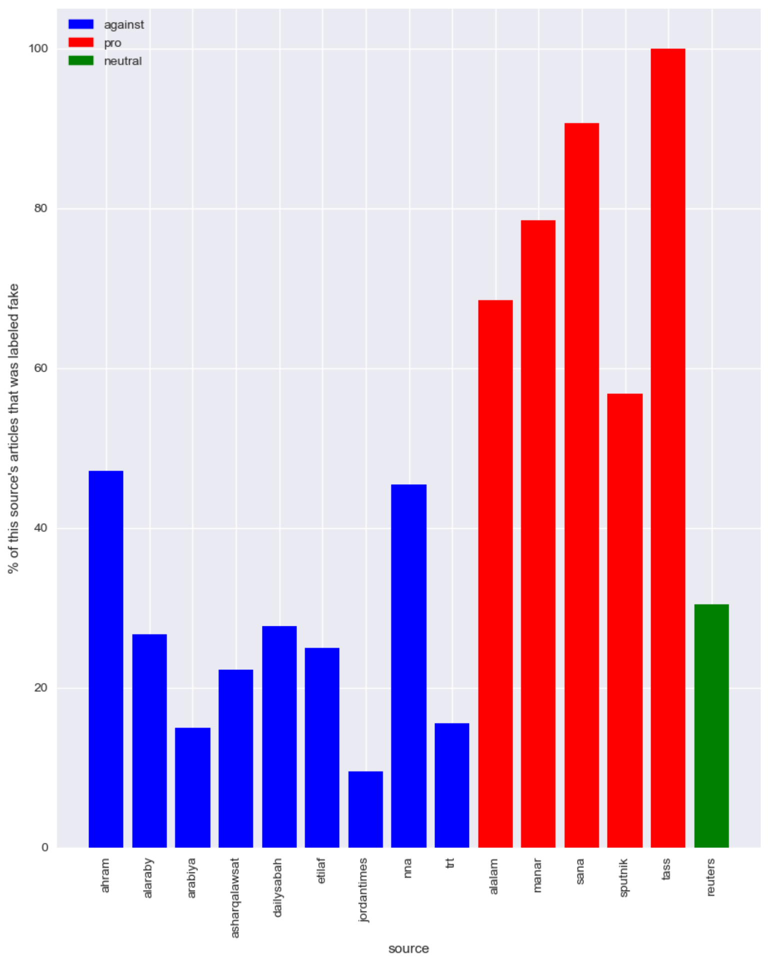


Figure 5.8: Plot Showing the % of Articles Labeled Fake for Each Source where % = nb_fake_per_source / nb_total_per_source

5.3 Sectarian Language

The sectarian language feature of a particular article is the frequency of the sectarian words in this article. In order to detect the number of sectarian words used in an article, we first built a sectarian language lexicon. In order to make sure that the sectarian language lexicon we built actually detects the sectarian language that is used in the reporting of the war events of the Syrian war, we built the lexicon using articles reporting about the Syrian war. Some of these articles were reporting war events, others were discussing the situation, attitude, and interests of a certain sect/religion in the Syrian war. We also included articles about the religions and sects to enrich our lexicon with all different forms of language that revolve around certain religions. The following are the articles we obtained for this purpose:

- Wilayat al-Faqih project aimed at destroying Arabs - Experts
- Iran Is Building a New Source of Shia Influence Inside Syria
- Iran repopulates Syria with Shia Muslims to help tighten regime's control
- 'Takfiri' Crimes against Islam in Syria Escalate
- No Muslims among the Takfiri Groups In Syria
- Inside the mind of the takfiris in Syria
- The Ahrar al Sham Movement: Syria's Local Salafists
- Salafism Vs. Wahhabism: Qatar and Saudi Arabia's Proxy War Rages In Syria Thanks To US Militarism
- You Can't Understand ISIS If You Don't Know the History of Wahhabism in Saudi Arabia
- Why so Much Hate? Closer Look at Erdogan's 'Kurdophobia'
- The Trouble with Turkey: Erdogan, ISIS, and the Kurds
- Why Turkey Sees the Kurdish People as a Bigger Threat than ISIS
- The role of Christian militias in Syria goes largely ignored
- Assad's Shabiha: "Starvation or Submission" to Civilians
- What is Wilayat al-Faqih?
- Guardianship of the Islamic Jurist

- Shia–Sunni relations
- What is Wahhabism? The reactionary branch of Islam from Saudi Arabia said to be ‘the main source of global terrorism’
- Wahhabism
- Salafi movement

Once we collected the above articles, we extracted the words that are sectarian out of them and built our sectarian language lexicon from these words. We believe that our sectarian language lexicon contains all of the sectarian words that are used in reporting not only the war in Syria but a lot of events around the middle east.

5.3.1 Examples of Sectarian Words in Our Dataset

The following are sentences extracted from our dataset that contained sectarian words. The words in italics are the sectarian words, based on our sectarian language lexicon.

1. The campaign backed by Assad ally Russia has captured large swathes of territory from the *terror* group but they have launched deadly counterattacks on regime positions.
2. Meanwhile in the east a Syrian Kurdish news agency says clashes have erupted again between YPG and Syrian pro-regime *militias* in the northern Syrian city of Hasakeh where two groups have shared control of the city since the early years of the Syrian civil war.
3. Damascus: a bomb claimed by the terrorist group Nusra Front tore apart a bus carrying Lebanese *Shi'ite* Muslim *pilgrims*
4. The army units meantime thwarted an attempt by the *Takfiri* terrorists to penetrate into a Syrian army base in Jabal al-Rahmalia region in Northern Lattakia.
5. It did not say what was in the barrels but appeared to suggest that some sort of chemical agent was inside and supplied by Saudi Arabia the region's *Sunni Muslim* power and a staunch supporter of Syria's Sunni-led revolt
6. A bomb has exploded inside a mosque in a suburb of Damascus killing a *Sunni Muslim* cleric as fierce fighting rages on in other parts of the country.
7. *Sunni Muslim* preacher Sheikh Mohammad Said Ramadan al-Buti - an outspoken supporter of President Bashar al-Assad - was killed along with at least 41 others when a suicide bomber struck a Damascus mosque.

8. In another development Syrian tribal leaders are in secret talks with UN special envoy for Syria and a senior US general to form a coalition similar to the so-called "*Sunni* Awakening" during the US occupation of Iraq according to The Independent.

9. Rebels say it amounts to forced displacement of Assads opponents from Syrias main urban centers in the west of the country and engenders demographic change because most of the opposition and Syrias population are *Sunni*.

10. The strike targeted the mainly *Druze* region of al-Hadr in the Quneitra province on the Golan Heights the Britain-based watchdog added.

5.3.2 Extracting the Feature

For each article in our dataset, we calculated the frequency of the sectarian words by dividing the number of sectarian words in the article by the total number of words of the article. We do so using the following formula:

$$sectarian\_freq = \frac{nb\_sectarian\_words}{nb\_total\_words}$$

5.3.3 Exploratory Analysis

As discussed in section 5.3, the sectarian language feature is equal to the frequency of the sectarian words in the article. The plots in figures 5.9 and 5.10 show the distribution of the sectarian frequencies in articles labeled true/fake. The plots show that for the articles labeled fake, 95.2% have sectarian language frequencies between 0 and 0.08 and 4.8% have sectarian language frequencies between 0.07 and 0.15. Whereas for the articles labeled true, 97.9% have sectarian language frequencies between 0 and 0.04 and 2.1% have sectarian language frequencies between 0.05 and 0.13. Therefore, the articles labeled true have lower sectarian frequencies than the articles labeled fake (lower cut-off).

5.4 Consistency Score

This feature determines how consistent an article claim is with respect to other articles reporting the same event from the same source-category. It is equal to the average distance of an article's claims from other articles from the same category that are reporting on the same event.

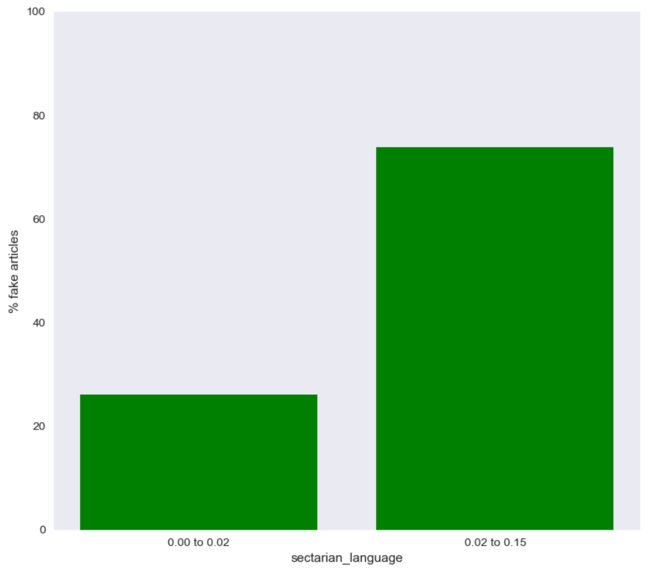


Figure 5.9: Plot Showing the Distribution of Sectarian Language Frequencies in Articles Labeled Fake

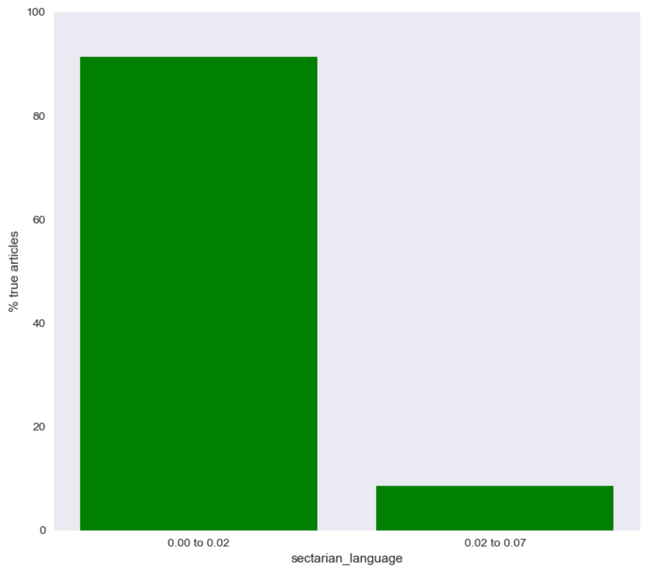


Figure 5.10: Plot Showing the Distribution of Sectarian Language Frequencies in Articles Labeled True

5.4.1 Extracting the Feature

In order to extract the consistency We first grouped our sources into three categories: proSyrian regime, against Syrian regime, and neutral.

The sources in our dataset that are against the Syrian regime are the following:

- Arabiya
- Jordan Times
- Al Ahram
- Asharq Al Awsat
- Lebanese National News Agency
- Etilaf
- Al Araby
- TRT
- Daily Sabah

The sources in our dataset that are pro-Syrian regime are the following:

- SANA
- Al Alam
- Al Manar
- Sputnik
- TASS

The source in our dataset that is neutral is Reuters.

In order to calculate the consistency score for a certain article X, we first mapped X to all other articles in our dataset that are from the same source-category as X (pro, against, or neutral) and are reporting the same event. We did so by extracting the articles in our dataset that have the same date as X and the same sourcecategory as the source of X. We then calculated the Gower distance of the claims between X and each of the extracted articles. The Gower distance is calculated based on the following claims: nb_civilians, nb_children, nb_women, nb_noncivilians, actor, and cause of death, similar to the approach we did in mapping an article to its VDC event 4. The consistency score feature value of X is then the average of the Gower distances between X and each of the mapped articles. This is calculated based on the following formula:

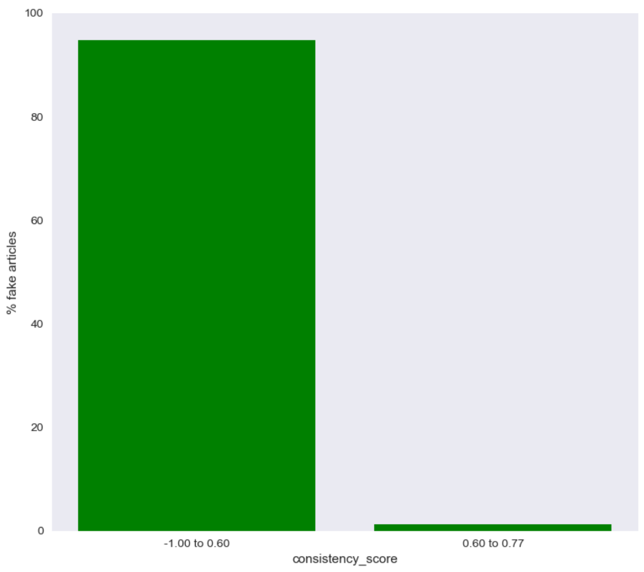


Figure 5.11: Plot Showing the Distribution of the Consistency Scores in Articles Labeled Fake

$$consistency\_score\_x = AVERAGE(GOWER\_DISTANCE(x, y))|y \in Y$$
$$\text{Where } Y \text{ is the set of articles where: } Y\_category = x\_category \text{ and } Y\_date = x\_date$$

If the article was not matched with other articles, i.e. no articles in our dataset from the same source-category reported the same event, we set this feature to -1.

5.4.2 Exploratory Analysis

As discussed in section 5.4, the consistency score feature is equal to the average Gower distance between the article claims and other articles from the same source-category reporting the same event. If the article was not matched with any event, this feature is -1. The plots in figures 5.11 and 5.12 show that for the articles labeled fake, 95.8% have consistency score value between 0 and 0.75 and 4.2% have negative consistency score value (not matched with any other article in our dataset reporting the same event). Whereas for the articles labeled true, 98% have consistency score value between -1 and 1 and 1.2% have consistency score values greater than 1. Therefore, the articles labeled true have lower consistency scores than the articles labeled fake (lower cut-off).

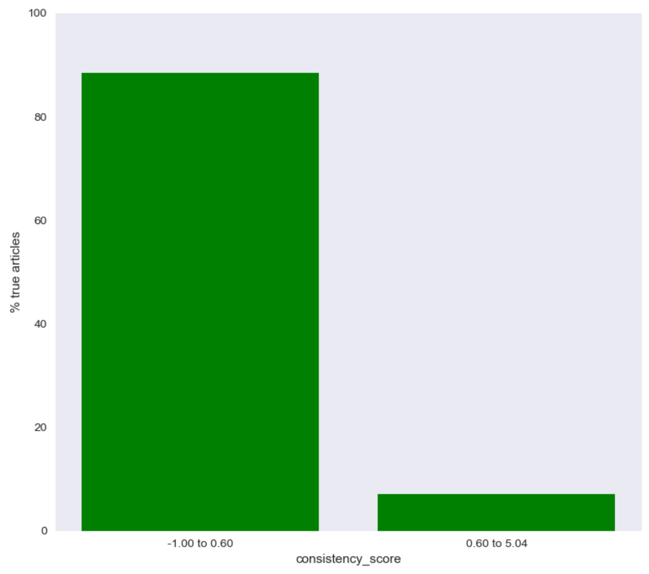


Figure 5.12: Plot Showing the Distribution of the Consistency Scores in Articles Labeled True

5.5 Description of Sources Quoted in Reporting the News

This feature describes the quoted sources in the article's claim. When reporting a certain event, articles should quote a source for their claims. A lot of articles quote 'local media sources', 'a source claimed', 'activists say', etc... which are all not real attribution to a person or an organization for the article to back up their claims properly. The values of this feature can be one of the following:

- Value 0: no source, the article just claims the event and does not provide any sources for its claim.

- Value 1: no real attribution: article quotes 'sources', 'activists', etc... without naming the source/activists (e.g. local media sources, a source in the army, etc...)

- Value 2: real attribution (named source): The article specifies the name of the organization as a source or the name of the person who is its source (e.g. WHO organization, name of the activist, etc...)

5.5.1 Extracting the Feature

For each article in our dataset, we use the following steps in order to set the value of its quoted sources feature:

- We first extract the sentences that contain report verbs (from the report verbs lexicon used in calculating the report verbs feature)
- No quoted sources => value 0
- Using Stanford Dependency Parser, we find the subject of the verb (e.g. if sentence is activists say, subject is activists, etc...)
- Once we have the subject of the report verb:
- In order to decide if the article's source is an organization, a specific name, or neither, we use Stanford's Named Entity Recognizer to classify the subject of the report verb.
- Stanford's classifier classifies the subject as either 'Organization', 'Person', 'Location', or 'Other'. Based on the output of Stanford's classifier, we decide if they gave real attribution (quoted an organization or a person) => value 3
- if the source is 'other', the value of the feature is no real attribution => value 1

5.5.2 Exploratory Analysis

As discussed in section 5.5, the value of the quoted sources can be one of the following: 0 if no sources mentioned at all, 0.5 if no real attribution, and 1 if the source quoted is an organization or a person. The plots of figures 5.13 and 5.14 show the distribution of the quoted sources feature in articles labeled true/fake. The plots show that for the articles labeled fake, 13.8% have quoted sources value between 0 and 0.4 and 86% have quoted sources value between 0.4 and 1. Whereas for the articles labeled true, 31% have quoted sources value between 0.5 and 0.8 and 69% have quoted sources value between 0.8 and 1. Therefore, the articles labeled true have higher quoted sources value than the articles labeled fake (higher starting value and higher cut-off).

5.6 Assertive Verbs

This feature is one of the stylistic features from [6] that we used in our feature extraction. According to [6], assertive verbs capture the degree of certainty to

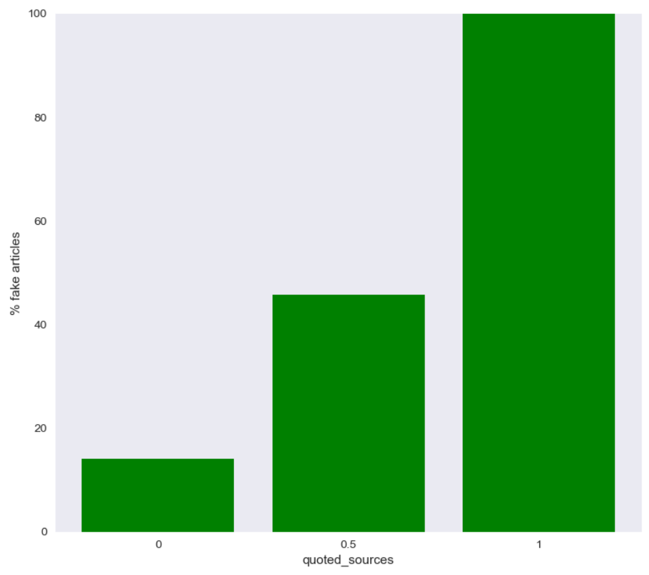


Figure 5.13: Plot Showing the Distribution of the Quoted Sources Feature in Articles Labeled Fake

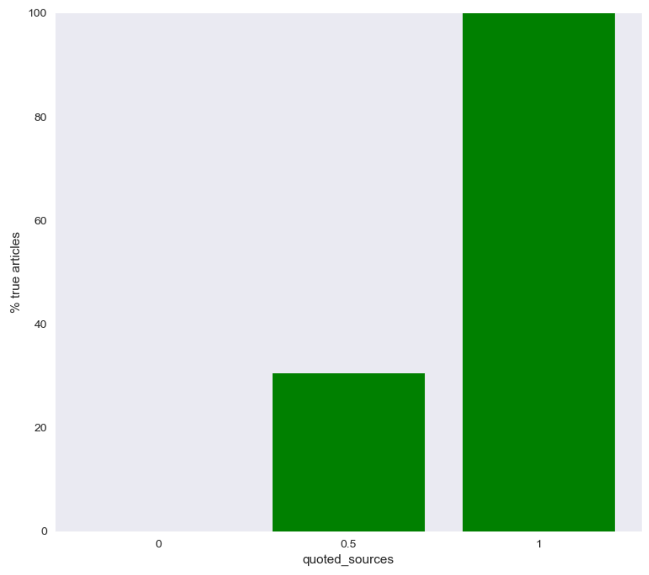


Figure 5.14: Plot Showing the Distribution of the Quoted Sources Feature in Articles Labeled True

which a proposition holds. The authors suggest an assertive verbs lexicon. Examples of these words include: guess, seem, appear, figure, believe, expect, etc... We use this lexicon in order to calculate the frequency of assertive verbs in our articles as our assertive verbs feature.

5.6.1 Examples of Assertive Verbs in Our Dataset

The following are sentences extracted from our dataset that contained assertive verbs. The words in italics are the assertive verbs, based on our assertive verbs lexicon.

1. The attack is *believed* to be among the deadliest in the city for years the British-based Syrian Obervatory for Human Rights said
2. The death toll is *expected* to rise because of the number of people seriously injured added the Observatory
3. Victims of a suspected chemical attack in Syria *appeared* to show symptoms consistent with reaction to a nerve agent the World Health Organization said on Wednesday.
4. The document *alleges* several cases of the Syrian regime using the method on its own people.
5. Al-Araby cannot independently *verify* any of the cases presented in the joint report.
6. Opposition forces say regime launched surprise attack to cut off rebel-held areas of Aleppo hours after the UN said it had *agreed* in principle to a six-week ceasefire.
7. Syrian rebels *claim* to have repelled a surprise regime attack north of Aleppo and taken prisoners launched after the regime agreed in principle to a UN-brokered six-week ceasefire.
8. The Reuters news agency *reported* that at least 70 pro-regime fighters and more than 80 rebels were killed as rebels countered the attack on Tuesday and Wednesday which was intended to cut supply lines to the city.
9. Chlorine and mustard gas which are also *believed* to have been used in the past in Syria.
10. US airstrikes target the IS The Syrian Observatory for Human Rights reported the death of two senior Islamic State group (IS) leaders in a *suspected* US airstrike on Monday

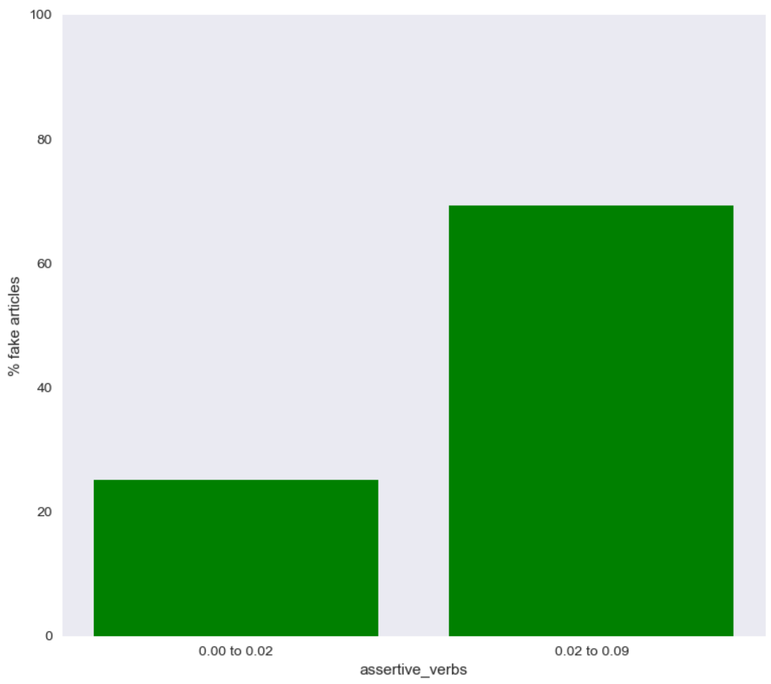


Figure 5.15: Plot Showing the Distribution of Assertive Verbs Frequencies in Articles Labeled Fake

5.6.2 Extracting the Feature

For each article in our dataset, we calculated the frequency of the assertive verbs by dividing the number of assertive verbs in the article by the total number of words of the article. We do so using the following formula:

$$assertive\_freq = \frac{nb\_assertive\_verbs}{nb\_total\_words}$$

5.6.3 Exploratory Analysis

As discussed in section 5.6, the assertive verbs feature is equal to the frequency of the assertive verbs in the article. The plots in figures 5.15 and 5.16 show the distribution of the assertive verbs frequencies in articles labeled true/fake. The plots show that for the articles labeled fake, 78.8% have assertive verbs frequencies between 0 and 0.04 and 21.2% have assertive verbs frequencies between 0.04 and

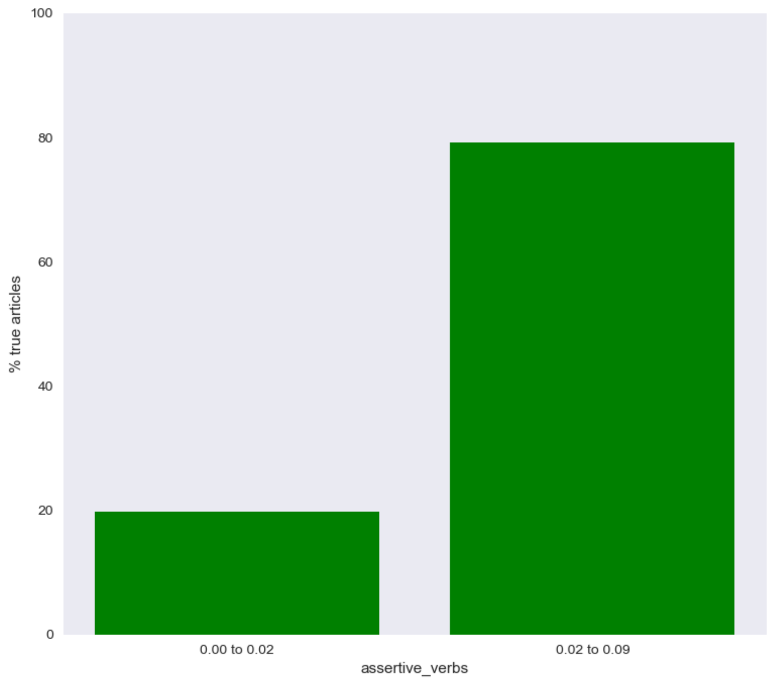


Figure 5.16: Plot Showing the Distribution of Assertive Verbs Frequencies in Articles Labeled True

0.08. Whereas for the articles labeled true, 76.5% have assertive verbs frequencies between 0 and 0.04 and 23.5% have assertive verbs frequencies between 0.04 and 0.08. Therefore, both the articles labeled fake and articles labeled true have similar assertive verbs frequencies distribution. However, more articles labeled true have assertive verb frequencies greater than the cut-off (23.5%) than articles labeled fake (21.2%).

5.7 Factive verbs

This feature is one of the stylistic features from [6] that we used in our feature extraction. According to [6], factive verbs presuppose the truth of a proposition in a sentence. The authors suggest a factive verbs lexicon. Examples of these words include: note, notice, observe, know, etc... We use this lexicon in order to calculate the frequency of factive verbs in our articles as our factive verbs feature.

5.7.1 Examples of Factive Verbs in Our Dataset

The following are sentences extracted from our dataset that contained factive verbs. The words in italics are the factive verbs, based on our factive verbs lexicon.

1. US officials on Tuesday said they remained skeptical of the IS claims Mueller died in an air strike *noting* there had been no evidence of civilians at that site before it was targeted.

2. Everyone *knows* Kobane it's where the Kurds stopped IS

3. The BBC quoted a former commanding officer of the British army's Joint Chemical Biological Radiological Nuclear Regiment Hamish de Bretton-Gordon as saying the footage would be very difficult to stage-manage but that samples taken from the scene would be *reveal* if chemical weapons had been deployed

4. The Britain-based monitor said that villagers had *discovered* the bodies when they returned to their homes after the regime forces withdrew a day later.

5. The ways and means of terrorism changed over time and it *makes sense* to take account of that that applies here and it doesn't apply only here.

6. Syria chemical weapons attacks since 2011 The Syrian government *acknowledges* for the first time that it has chemical weapons and threatens to use them in the event of military operations by Western countries but not against its own population.

7. Despite all the *odds* the Syrian people will not hesitate to uphold the rights to freedom justice and dignity.

8. We hope that *relevant* parties can continue communications and coordination and hold deep consultations so as to resolve the relevant issue in a peaceful way he added.

9. The chemical raids come as the Syrian army is making progress in operations to retake Aleppo from militants who are *seeing* the noose tightening around them in the areas which they control.

10. As far as we *know* from the information we've had from the defense ministry those in the helicopter died they died heroically because they were trying to move the aircraft away to minimize victims on the ground.

5.7.2 Extracting the Feature

For each article in our dataset, we calculated the frequency of the factive verbs by dividing the number of factive verbs in the article by the total number of words of the article. We do so using the following formula:

$$factive\_freq = \frac{nb\_factive\_verbs}{nb\_total\_words}$$

5.7.3 Exploratory Analysis

As discussed in section 5.7, the factive verbs feature is equal to the frequency of the factive verbs in the article. The plots in figures 5.17 and 5.18 show the distribution of the factive verbs frequencies in articles labeled true/fake. The plots show that for the articles labeled fake, 96.1% have factive verbs frequencies between 0 and 0.01 and 3.9% have factive verbs frequencies between 0.01 and 0.02. Whereas for the articles labeled true, 98.5% have assertive verbs frequencies between 0 and 0.01 and 1.5% have factive verbs frequencies between 0.01 and 0.02. Therefore, both the articles labeled fake and articles labeled true have similar factive verbs frequencies distribution. However, more articles labeled fake have factive verb frequencies greater than the cut-off (3.9%) than articles labeled fake (1.5%).

5.8 Hedges

This feature is one of the stylistic features from [6] that we used in our feature extraction. According to [6], hedges soften the degree of commitment to

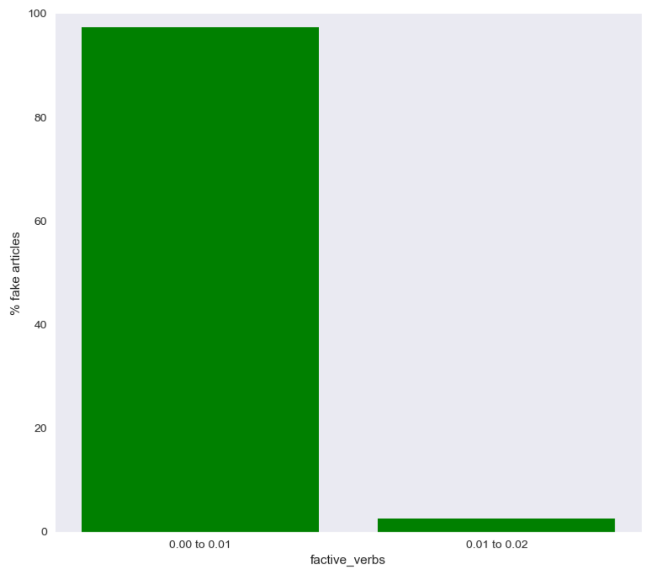


Figure 5.17: Plot Showing the Distribution of Factive Verbs Frequencies in Articles Labeled Fake

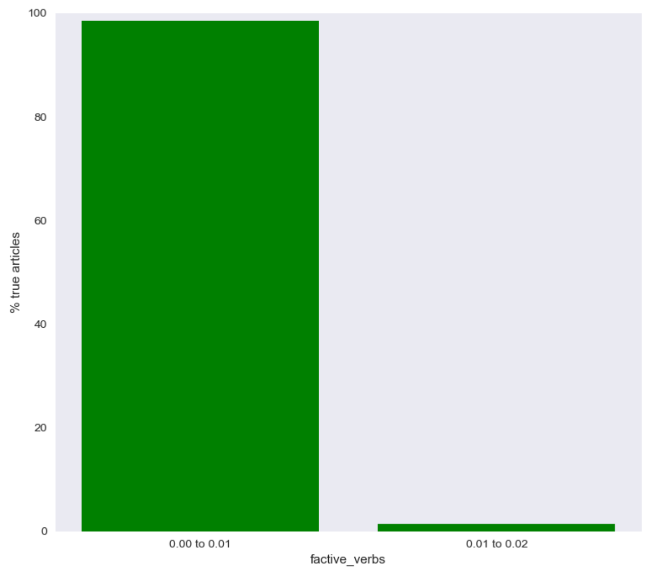


Figure 5.18: Plot Showing the Distribution of Factive Verbs Frequencies in Articles Labeled True

a proposition. The authors suggest a hedges lexicon. Examples of these words include: almost, apparently, appear, around, etc... We use this lexicon in order to calculate the frequency of hedges in our articles as our hedges feature.

5.8.1 Examples of Hedges in Our Dataset

The following are sentences extracted from our dataset that contained hedges. The words in italics are the hedges, based on our hedges lexicon.

1. Observatory director Rami Abdel Rahman said the blasts *appeared* to be coordinated

2. On Thursday *around* 120 opposition fighters and their families were evacuated from the last opposition-held district of Homs

3. The WHO said it was *likely* that some kind of chemical was used in the attack because sufferers had no apparent external injuries and died from a rapid onset of similar symptoms including acute respiratory distress.

4. Missile attack on one of its air bases had killed six people and caused extensive damage adding that it *would* respond by continuing its campaign to "crush terrorism" and restore peace and security to all of Syria.

5. Officials said the military fired dozens of cruise missiles against the base in response to the *suspected* gas attack in a rebel-held area that Washington has blamed on Assad's forces.

6. There was no immediate *claim* of responsibility for the attack which pro-Damascus media said was carried out by a suicide car bomber.

7. Residents said the capital was quiet on Friday and that the mortar and rocket fire *appeared* to be over.

8. In an interview broadcast by the BBC Syrian President Bashar al-Assad confirmed there was no cooperation with the coalition members of which he accused of backing "terrorism" - an *apparent* reference to their support for other rebel groups fighting to overthrow him.

9. *Around* 100 fighters on both sides have been killed according to monitoring group the Syrian Observatory for Human Rights.

10. In the surrounding countryside the situation is *largely* the reverse with rebels controlling much of the area west of the city and regime forces much of the east.

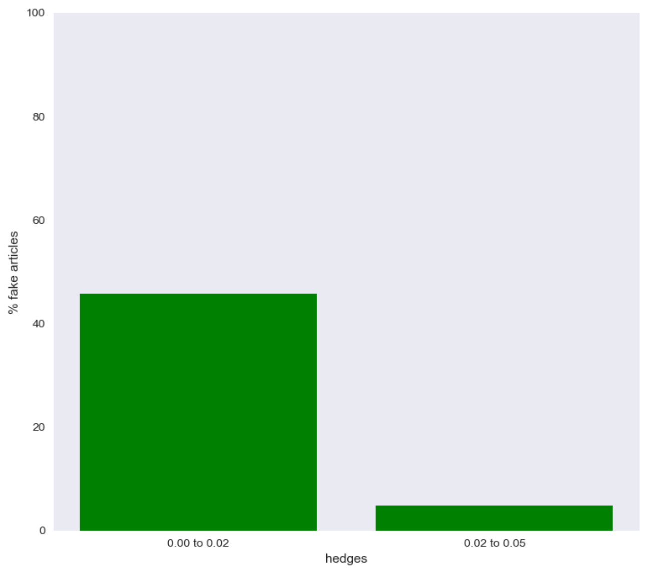


Figure 5.19: Plot Showing the Distribution of Hedges Frequencies in Articles Labeled Fake

5.8.2 Extracting the Feature

For each article in our dataset, we calculated the frequency of hedges by dividing the number of hedges in the article by the total number of words of the article. We do so using the following formula:

$$hedges\_freq = \frac{nb\_hedges}{nb\_total\_words}$$

5.8.3 Exploratory Analysis

5.8.4 Hedges

As discussed in section 5.8, the hedges feature is equal to the frequency of the hedges in the article. The plots in figures 5.19 and 5.20 show the distribution of the hedges frequencies in articles labeled true/fake. The plots show that the majority of both true and fake articles have hedges frequencies closer to zero. The plots show that for the articles labeled fake, 96.5% have hedges frequencies between 0 and 0.02 and 3.5% have hedges frequencies between 0.02 and 0.04. Whereas for the articles labeled true, 97.9% have hedges frequencies between 0

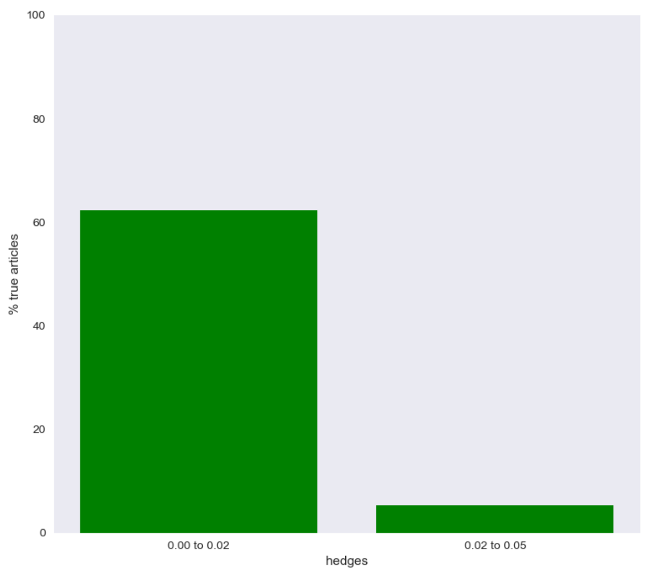


Figure 5.20: Plot Showing the Distribution of Hedges Frequencies in Articles Labeled True

and 0.02 and 2.1% have hedges frequencies between 0.02 and 0.04. Therefore, both articles labeled fake and articles labeled true have low hedges frequencies. However, more articles labeled fake have factive verb frequencies greater than the cut-off (3.5%) than articles labeled fake (2.1%).

5.9 Implicative Verbs

This feature is one of the stylistic features from [6] that we used in our feature extraction. According to [6], implicative verbs trigger presupposition in an utterance. For example, usage of the word complicit indicates participation in an activity in an unlawful way. The authors suggest an implicative verbs lexicon. Examples of these words include: bother, dare, neglect, force, fail, etc... We use this lexicon in order to calculate the frequency of implicative verbs in our articles as our implicative verbs feature.

5.9.1 Examples of Implicative Verbs in Our Dataset

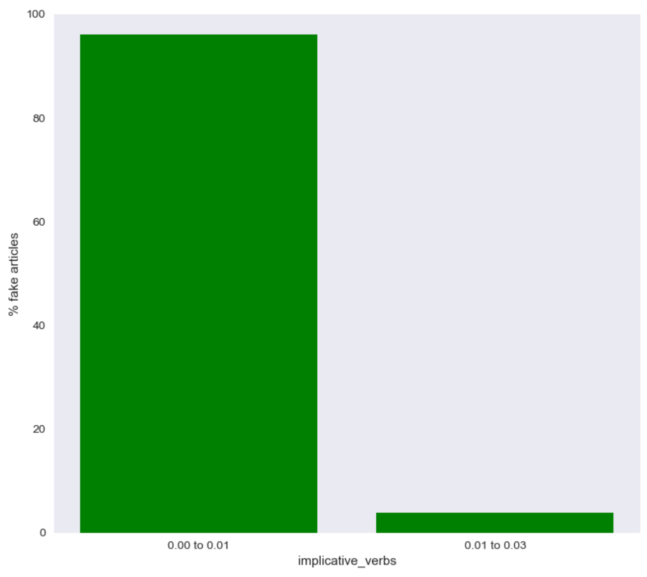
The following are sentences extracted from our dataset that contained implicative verbs. The words in italics are the implicative verbs, based on our implicative verbs lexicon.

1. Meanwhile fighting between Islamic State militants and a Kurdish-Arab alliance troops has *forced* 13000 residents to flee the IS-bastion city of Manbij
2. Regime forces continued their siege on opposition-held areas of Aleppo after a rebel advance *failed* with new fears of impending hunger and disease for Syrians in the city
3. The United States has said the deaths were *caused* by sarin nerve gas dropped by Syrian aircraft.
4. He said the attack was a form of "support for the armed terrorist groups and it is an *attempt* to weaken the capabilities of the Syrian Arab Army to combat terrorism".
5. The White House said IS had sent Mueller's distraught family a "private message" that was "authenticated" by intelligence *allowing* them to confirm her death.
6. 'There is no dialogue' Syria has grudgingly accepted the air strikes against IS but has repeatedly criticized the coalition for *failing* to coordinate with it.
7. It says the raids *cannot* defeat IS unless the international community starts cooperating with Syrian troops on the ground.
8. 'Restoring security' The official Syrian news agency Sana quoted a military source on Wednesday as saying the army was *able* to "restore security" to Sanei Tal al-Hawa and Tal-Arous in the southwestern countryside of Damascus Tal Meri west of Damascus and al-Danaji and Tal Antar near Deir al-Adas.
9. He had not *managed* to get a commitment by the rebels to a trial ceasefire before heavy fighting broke out Tuesday.
10. But the IS has encroached gradually in the province and made a series of *attempts* to enter the city finally *succeeding* in an operation that began on 25 June.
11. I would never *allow* those IS savages to violate my land.

5.9.2 Extracting the Feature

For each article in our dataset, we calculated the frequency of implicative verbs by dividing the number of implicative verbs in the article by the total number of words of the article. We do so using the following formula:

Figure 5.21: Plot Showing the Distribution of Implicative Verbs Frequencies in Articles Labeled Fake



$$implicative\_freq = \frac{nb\_implicative\_verbs}{nb\_total\_words}$$

5.9.3 Exploratory Analysis

The implicative verbs feature is equal to the frequency of the implicative verbs in the article. The plots in figures 5.21 and 5.22 shows the distribution of the implicative verbs frequencies in articles labeled true/fake. The plots show that for the articles labeled fake, 97.7% have implicative verbs frequencies between 0 and 0.01 and 2.3% have implicative verbs frequencies between 0.01 and 0.02. Whereas for the articles labeled true, 88.9% have implicative verbs frequencies between 0 and 0.008 and 11.1% have implicative verbs frequencies between 0.008 and 0.016. Therefore, the articles labeled true have lower implicative verbs frequencies than the articles labeled fake (lower cut-off).

5.10 Report Verbs

This feature is one of the stylistic features from [6] that we used in our feature extraction. According to [6], report verbs emphasize the attitude towards the

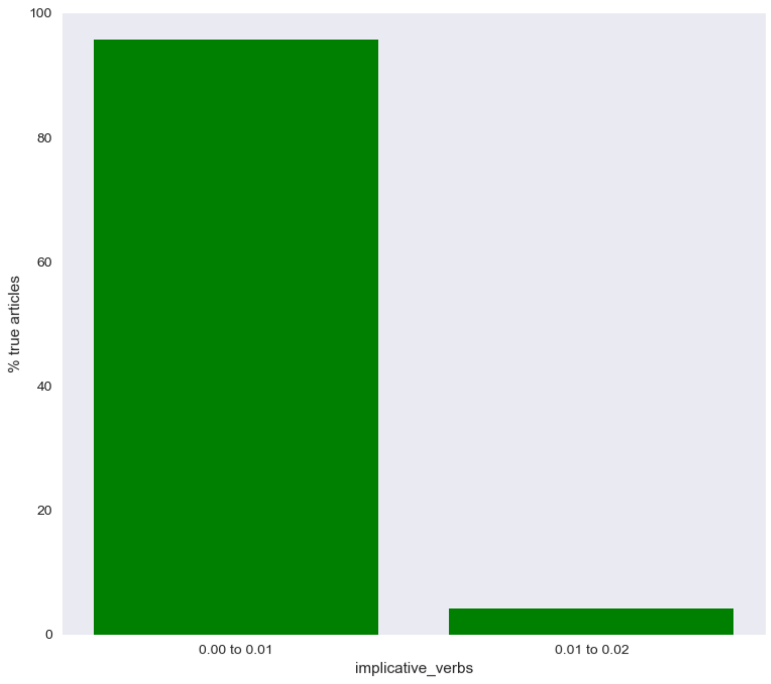


Figure 5.22: Plot Showing the Distribution of Implicative Verbs Frequencies in Articles Labeled True

source of the information. The authors suggest a report verbs lexicon. Examples of these words include: accuse, acknowledge, add, admit, advise, agree, etc... We use this lexicon in order to calculate the frequency of report verbs in our articles as our report verbs feature.

5.10.1 Examples of Report Verbs in Our Dataset

The following are sentences extracted from our dataset that contained report verbs. The words in italics are the report verbs, based on our report verbs lexicon.

1. However subsequent reports have disputed the group's claims *saying* that the downed aircraft had in fact belonged to the Syrian regime
2. An unnamed source in the Pentagon *warned* Reuters on Tuesday that Saudi Arabia would arm rebels with Grad rockets in order to "fight the Russians"
3. Assad's government has always *denied* responsibility for that attack.
4. But Russia a Syrian ally and China have repeatedly vetoed any United Nations move to sanction Assad or *refer* the situation in Syria to the International Criminal Court.
5. Missile attack on one of its air bases had killed six people and caused extensive damage *adding* that it would respond by continuing its campaign to "crush terrorism" and restore peace and security to all of Syria.
6. A statement from the army command *described* the attack on Friday as an act of "blatant aggression" saying it had made the United States "a partner" of Islamic State the Nusra Front and other "terrorist organizations".
7. We *hope* there are not many victims and martyrs.
8. He *said* the attack was a form of "support for the armed terrorist groups and it is an attempt to weaken the capabilities of the Syrian Arab Army to combat terrorism".
9. President Donald Trump said he ordered missile strikes against an airfield from which a deadly chemical weapons attack was launched this week *declaring* he acted in America's "national security interest" against Syrian President Bashar al-Assad.
10. President Donald Trump said he *ordered* missile strikes against an airfield from which a deadly chemical weapons attack was launched this week declaring he acted in America's "national security interest" against Syrian President Bashar al-Assad.

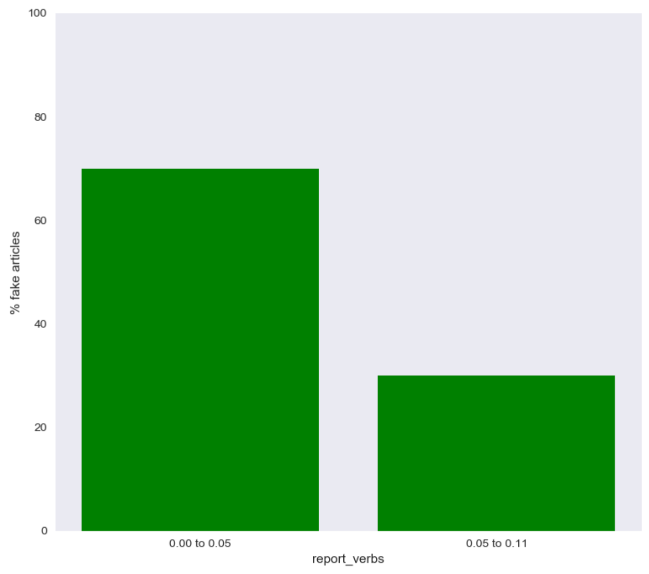


Figure 5.23: Plot Showing the Distribution of Report Verbs Frequencies in Articles Labeled Fake

5.10.2 Extracting the Feature

For each article in our dataset, we calculated the frequency of report verbs by dividing the number of report verbs in the article by the total number of words of the article. We do so using the following formula:

$$report\_freq = \frac{nb\_report\_verbs}{nb\_total\_words}$$

5.10.3 Exploratory Analysis

5.10.4 Report Verbs

As discussed in section 5.10, the report verbs feature is equal to the frequency of the report verbs in the article. The plots in figures 5.23 and 5.24 show the distribution of the report verbs frequencies in articles labeled true/fake. The plots show that for the articles labeled fake, 74.3% have report verbs frequencies between 0 and 0.05 and 25.7% have report verbs frequencies between 0.05 and 0.1. Whereas for the articles labeled true, 75.9% have report verbs frequencies between 0.01 and 0.06 and 24.1% have report verbs frequencies between 0.06 and

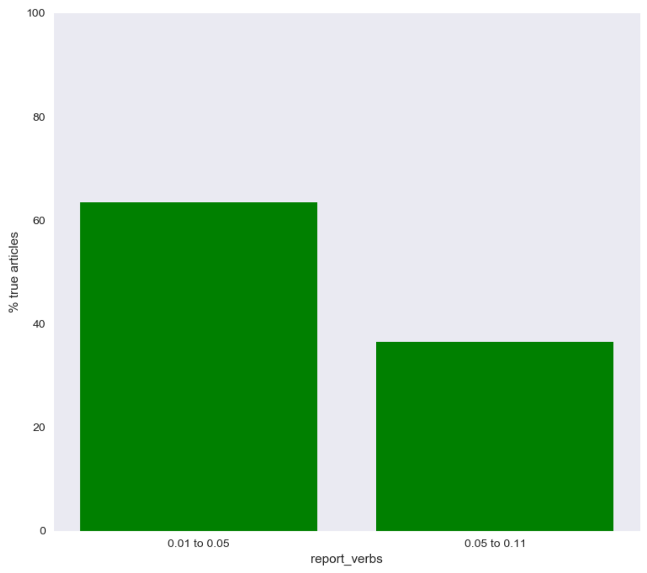


Figure 5.24: Plot Showing the Distribution of Report Verbs Frequencies in Articles Labeled True

0.1. Therefore, the articles labeled true have higher report verbs frequencies than the articles labeled fake (higher starting value and higher cut-off).

5.11 Subjectivity and Bias

This feature is one of the stylistic features from [6] that we used in our feature extraction. According to [6], news is supposed to be objective, writers should not convey their own opinions, feelings or prejudices in their stories. The authors suggest a subjectivity lexicon, a list of positive and negative opinionated words, an affective lexicon to detect subjective clues in articles, and a lexicon of bias-inducing words. Examples of these words include: abusive, bad, abuse, achievement, absurd, devastate, assault, etc... We use these lexicons in order to calculate the frequency of biased words in our articles as our bias feature.

5.11.1 Examples of Bias in Our Dataset

The following are sentences extracted from our dataset that contained biased words. The words in italics are the biased words, based on our bias and subjectivity lexicons.

1. Once a powerhouse of industry Aleppo has been *devastated* by years of

fighting between regime forces and a succession of rebel groups.

2. A surprise IS *assault* on Tal Abyad on Tuesday has been foiled and Kurdish and rebel fighters take back full control of the town.

3. It morphed into a conflict after the regime unleashed a *brutal* crackdown on dissent.

4. On Tuesday Syrian dictator Bashar al-Assad launched a *horrible* chemical weapons attack on innocent civilians.

5. Yahya Aridi spokesman for the Syrian opposition Geneva delegation said on Sunday it was now time for the Syrian government and the opposition to "get to the table and start talking about transition from *dictatorship* to freedom" in Geneva.

6. The Syrian civil war began in mid 2011 when protests against autocrat Bashar al Assad and his regime were suppressed *violently* in the context of the Arab Spring uprisings.

7. ISIS published a grisly video showing the *savage* execution of 17 young men from the town of Altibni on the first day of Eid.

8. It is violence by bureaucratic means rather than force of arms but it is just as *devastating*.

9. A *crime* against *humanity* has been committed and there is not any mention of accountability.

10. Fighters loyal to Damascus have said that rebels murdered a boy on Tuesday because of his Palestinian origins in a *despicable* act of revenge.

5.11.2 Extracting the Feature

For each article in our dataset, we calculated the frequency of biased words by dividing the number of biased words in the article by the total number of words of the article. We do so using the following formula:

$$bias\_freq = \frac{nb\_bias\_words}{nb\_total\_words}$$

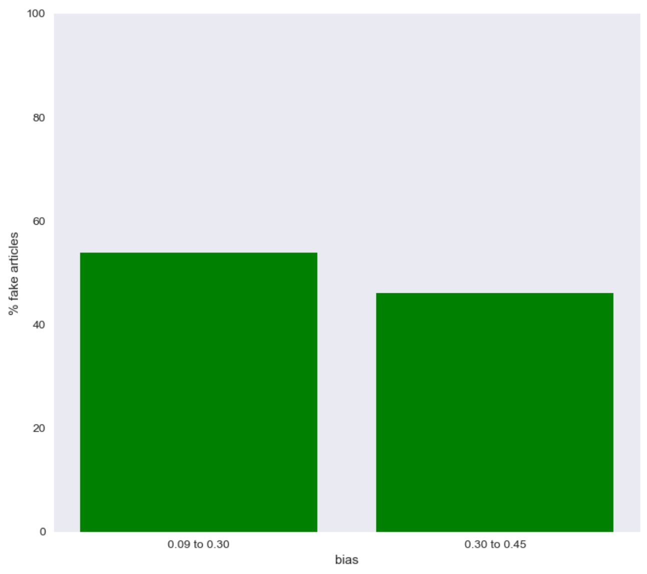


Figure 5.25: Plot Showing the Distribution of Bias Frequencies in Articles Labeled Fake

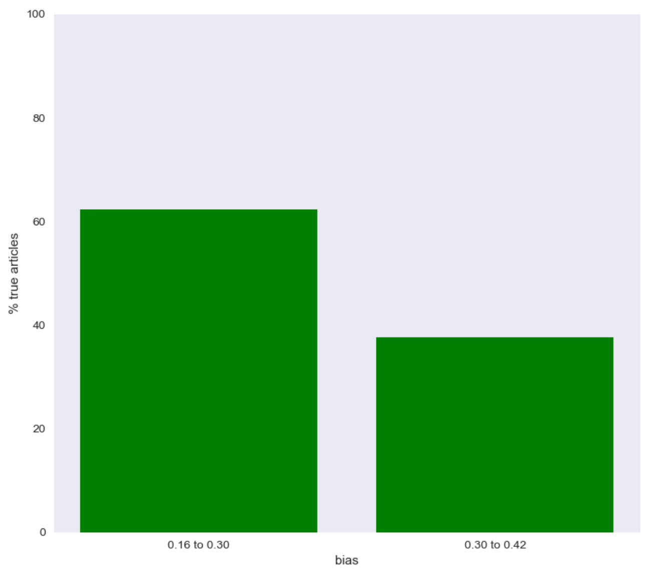


Figure 5.26: Plot Showing the Distribution of Bias Frequencies in Articles Labeled True

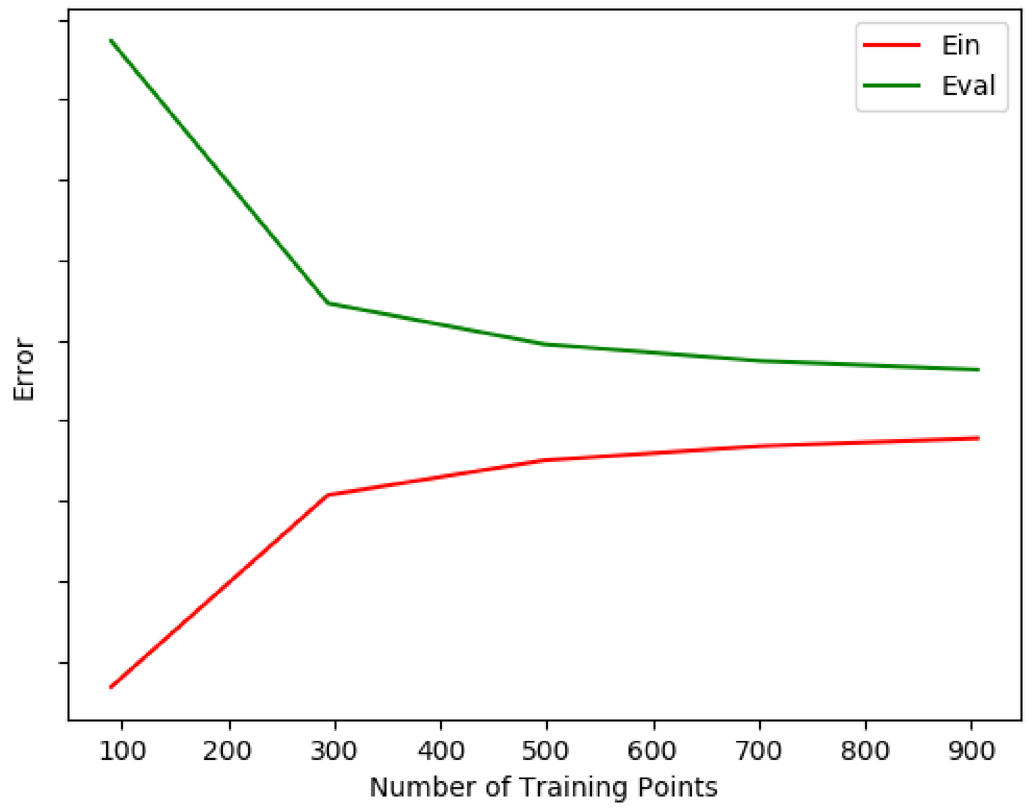


Figure 5.27: Learning Curve Using a Linear Regression Model

5.11.3 Exploratory Analysis

As discussed in section 5.11, the bias feature is equal to the frequency of the bias words in the article. The plots in figures 5.25 and 5.26 shows the distribution of the bias frequencies in articles labeled true/fake. The plots show that for the articles labeled fake, 32.5% have bias frequencies between 0 and 0.25 and 67.5% have bias frequencies between 0.25 and 0.45. Whereas for the articles labeled true, 53.9% have bias frequencies between 0 and 0.25 and 46.1% have bias frequencies between 0.25 and 0.45. Therefore, the articles labeled true have lower bias frequencies than the articles labeled fake (same cut-off but lower percentage above the cut-off for articles labeled true).

5.12 Learning Curve

We plot the learning curve of our training dataset using a simple linear regression model. Figure 5.27 shows that the gap between E_in and E_out is small and that both are very high. This means that the our model has high bias and that we need to do one of the following remedies and we need to use complex machine learning models.

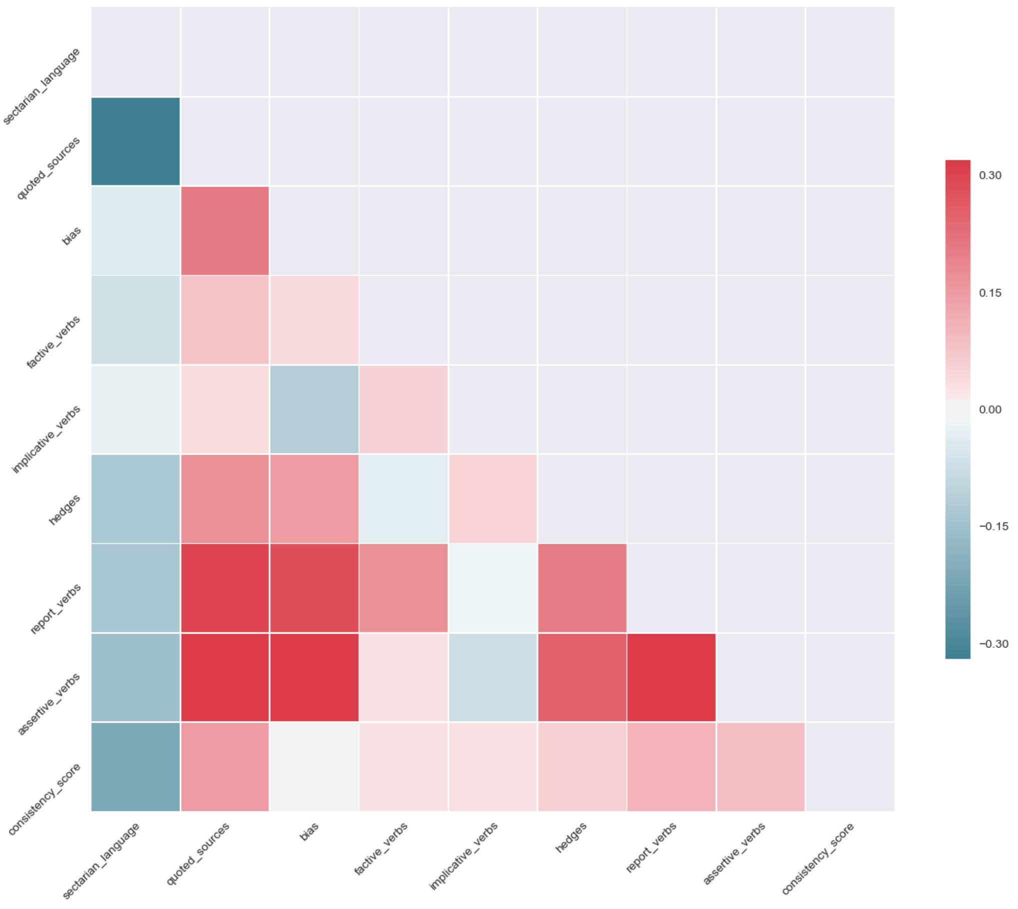


Figure 5.28: Plot Showing the Correlation HeatMap Of The Features

Chapter 6

Methodology

In this chapter, we describe the baseline and machine learning models that we tested and the hyper-parameter tuning that we performed on each model to get the highest accuracy. For each of the following models, the train dataset (80% of our entire dataset) was used for training, and the testing dataset (20% of our entire dataset) was used for testing. In the subsections to come, the reported best accuracy is the testing dataset accuracy.

6.1 Baseline Models

In this section, we describe the baseline models that we compare our machine learning models' performance to. We tested the baseline model that outputs the majority class for any input article. In our dataset, the majority class is the true class (53% of our articles were labeled true vs 47% articles were labeled fake). We also tested out Ripper and See5 Rules in order to visualize any rules that can be extracted through our features.

6.1.1 Ripper Rules

We tested Ripper model using R's Ripper implementation. This model resulted in an accuracy of 0.84 and the following Ripper rules:

1. If (sectarian_frequency >= 0.02) and (report_frequency >= 0.03) => label=fake

2. If (sectarian_frequency >= 0.03) => label=fake

3. If (quoted_sources == 0) => label = fake

4. If (quoted_sources <= 0.5) and (sectarian_frequency >= 0.01) and (bias >= 0.3) => label=fake

5. Else label=true

The above set of rules help us deduce the following about a given article:

1. If the sectarian language frequency is above 0.02 and the report verbs frequency is above 0.03 then the article is labeled fake.
2. If the sectarian language frequency is above 0.03, the article is labeled fake.
3. If the quoted sources feature is 0 (article does not quote any sources) then the article is labeled fake.
4. If the quoted sources feature is 0 (article does not quote any sources) or 0.5 (no real attribution) and the sectarian language frequency is above 0.01 and the bias is above 0.3, then the article is labeled fake.
5. Otherwise, the article is labeled true.

6.1.2 See5 Rules

We also tested the See5 model using R's C5 implementation. This model also resulted in an accuracy of 0.84 and the following See5 rules:

1. sectarian_frequency > 0.02 and factive_frequency <= 0 and report_frequency > 0.03 => class = fake
2. sectarian_frequency > 0.02 and implicative_frequency <= 0.001 and report_frequency > 0.03 => class = fake
3. sectarian_frequency > 0.02 and hedges <= 0.004 and report_frequency > 0.03 => class = fake
4. sectarian_frequency > 0.04 and bias > 0.23and consistency_score <= 0.2 => class = fake
5. quoted_sources == 0 => class = fake
6. sectarian_frequency > 0 and quoted_sources <= 0.5 and report_frequency > 0.03 and assertive <= 0.02 => class = fake
7. bias > 0.3 => class = fake
8. sectarian_frequency > 0 and sectarian_frequency <= 0.02 and quoted_sources > 0.5 => class = true
9. sectarian_frequency > 0 and sectarian_frequency <= 0.02 and quoted_sources > 0 and bias <= 0.3 and assertive > 0.02 => class = true

10. sectarian_frequency <= 0.02 and quoted_sources > 0.5 and factive_frequency > 0 and implicative_frequency > 0.001 => class = true

11. sectarian_frequency == 0 and quoted_sources > 0 and quoted_sources == 0.5 => class = true

12. sectarian_frequency == 0 and quoted_sources > 0 and bias <= 0.3 => class = true

13. factive_frequency > 0 and implicative_frequency > 0.001 and hedges > 0.004 and report_frequency > 0.03 => class = true

14. sectarian_frequency <= 0 and quoted_sources > 0.5 and report_frequency <= 0.03 => class = true

15. quoted_sources > 0 and bias > 0.23 and report_frequency <= 0.03 and consistency_score > 0.2 => class = true

16. sectarian_frequency <= 0.02 and quoted_sources > 0 and report_frequency <= 0.03 => class = true

17. sectarian_frequency <= 0.04 and quoted_sources > 0 and bias <= 0.3 and report_frequency <= 0.03 => class = true

18. sectarian_frequency <= 0.02 and quoted_sources > 0 and quoted_sources <= 0.5 and bias <= 0.3 and report_frequency <= 0.03 => class = true

19. sectarian_frequency > 0.02 and quoted_sources > 0 and bias <= 0.23 and report_frequency <= 0.03 => class = true

20. Default class: true

6.2 Machine Learning Model

For each of the following models, we perform hyper-parameter turning using grid search and a 10-fold cross validation on the training dataset. Once we find the model with the best hyper-parameters using the training dataset, we test the model on the test dataset and report the results.
Table 7.1 shows the area under the ROC curve, accuracy, recall, precision, f-score, percentage of test dataset labeled true, and percentage of test dataset labeled fake for each of the tested machine learning models and the baseline models. We also plot the confusion matrix (tables 7.2 to 7.5) and ROC curve (figures 7.1 to 7.4) of each of the tested models.

6.2.1 SVM

The best SVM model found was soft-margin SVM with the following hyper-parameters:

- kernel: rbf
- C: 3.7

6.2.2 Decision Tree Classifier

The best decision tree classifier had the following hyper-parameters:

- max_depth: 65
- max_features: 8
- min_samples_leaf: 3
- min_samples_split: 0.03

6.2.3 Polynomial Linear Regression

We also tested transformation of input features using sklearn's PolynomialFeatures which takes a degree n and transforms each of the features into a polynomial of degree n. E.g. if input feature is x and degree is 2, the transformed input feature becomes x^2.

The best degree for polynomial transformation was: degree = 2.

6.2.4 Naive Bayes

We also tested the Naive Bayes Model.

6.3 Feature Selection

6.3.1 SelectKBest

In order to find the most important features in our dataset, we make use of sklearn's SelectKBest, which scores the features by importance and select features according to the k highest scores. We test SelectKBest for K = 4 and find out that the most important 4 features in our case are: quoted sources, implicative verbs, sectarian language, and consistency score].

6.3.2 Feature Importance

In order to find the most important features in our dataset, we make use of sklearn's ExtraTreeClassifier, which ranks the features from the most important feature to the least important feature. The following are the list of features sorted from the most important feature to the least important feature:

1. sectarian language (0.418822)
2. quoted sources (0.120231)
3. bias (0.093525)
4. report verbs (0.077519)
5. assertive verbs (0.073748)
6. consistency score (0.072027)
7. feature hedges (0.055454)
8. implicative verbs (0.054356)
9. factive verbs (0.034318)

Figure 6.1 shows the bar plot of the importance of each feature.

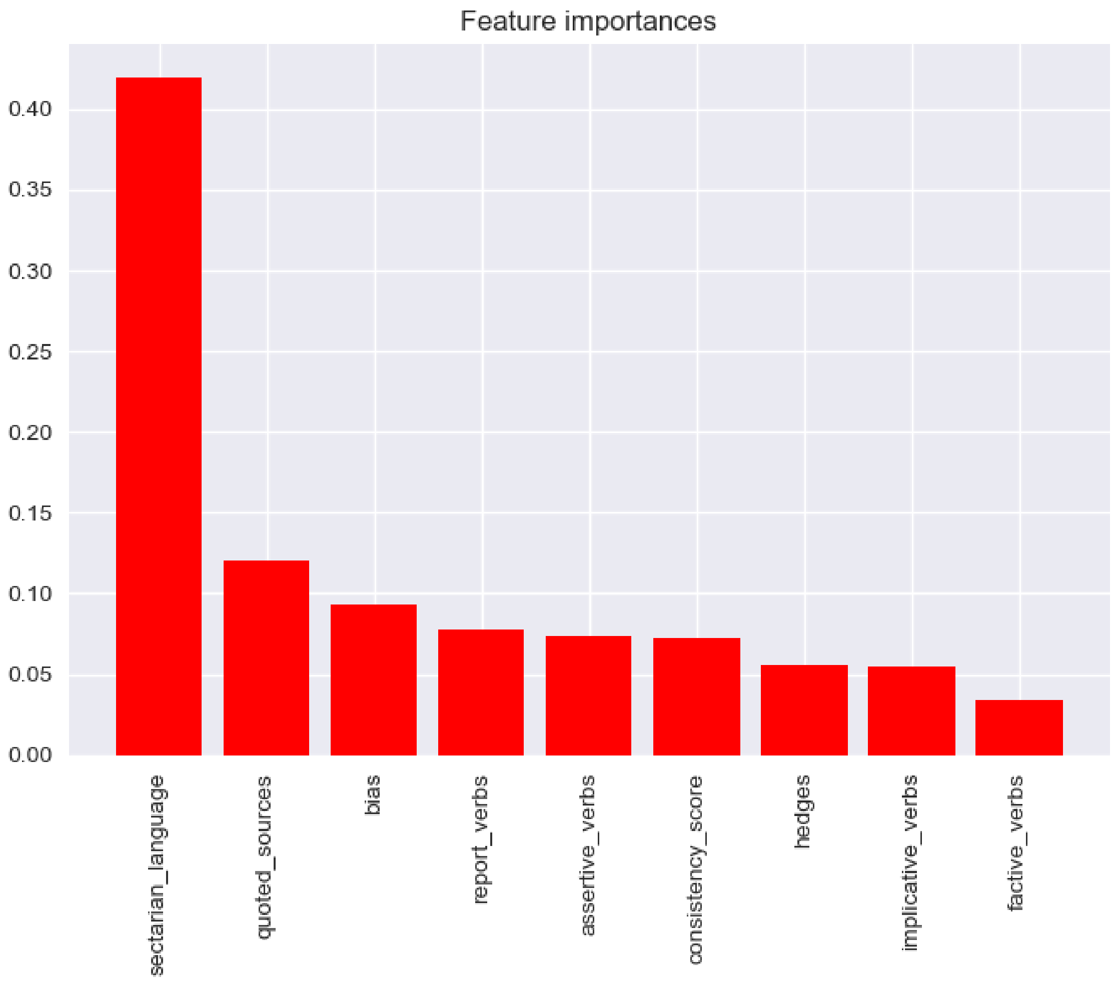


Figure 6.1: Feature Importance

Chapter 7

Results

7.1 Best Model

Table 7.1 shows the results of each of the tested machine learning models described in chapter 6. Based on this table, the best model with the highest accuracy, f-score, and area under the curve is the decision tree model. Tables 7.2 to 7.5 show the confusion matrix of each of our tested models. Figures 7.1 to 7.4 show the ROC curve of each of our tested models.

7.2 Testing Model on Related Work Dataset

In this section, we report the results of testing our best found model on the dataset used by [2]. As discussed in chapter 2, the dataset of [2] is made up of the Buzzfeed and Snopes dataset. The following are the results of testing our model on this dataset:

- Accuracy: 0.81
- F-score: 0.9

The results in this section show that our feature extraction approach and best found model generalizes well to other fake news datasets and not just the one we compiled or Syrian war news.

7.3 Testing The Quality of Our Dataset

In order to test the quality of the dataset of fake news articles we have built, we fit the model (the decision tree classifier) on the Buzzfeed and Snopes dataset and test it on the dataset that we have built. The following are the results:

- Accuracy 0.6

Table 7.1: Results of Tested Machine Learning Models

Model	AUC	Accuracy	Recall	Precision	F-score
SVM	0.86	0.87	0.89	0.88	0.89
Decision Tree	0.94	0.93	0.92	0.96	0.94
Polynomial Linear Reg.	0.82	0.83	0.86	0.85	0.85
Naives Bayes	0.84	0.85	0.89	0.85	0.87

Table 7.2: Confusion Matrix for the SVM Model

	Predicted Fake	Predicted True
Actual Fake	56	11
Actual True	10	84

Table 7.3: Confusion Matrix for the Decision Tree Classifier

	Predicted Fake	Predicted True
Actual Fake	64	3
Actual True	7	87

Table 7.4: Confusion Matrix for the Polynomial Linear Regression Model

	Predicted Fake	Predicted True
Actual Fake	53	14
Actual True	13	81

Table 7.5: Confusion Matrix for the Naives Bayes Model

	Predicted Fake	Predicted True
Actual Fake	53	10
Actual True	14	84

- F-score 0.7

This shows that training the model on the dataset we have built about the Syrian war allowed us to gain a higher accuracy and f-score on the Buzzfeed and Snopes datasets than when we used the Buzzfeed and Snopes dataset for training and our dataset for testing. This shows that our dataset allows us to train a more accurate model that generalizes to other fake news.

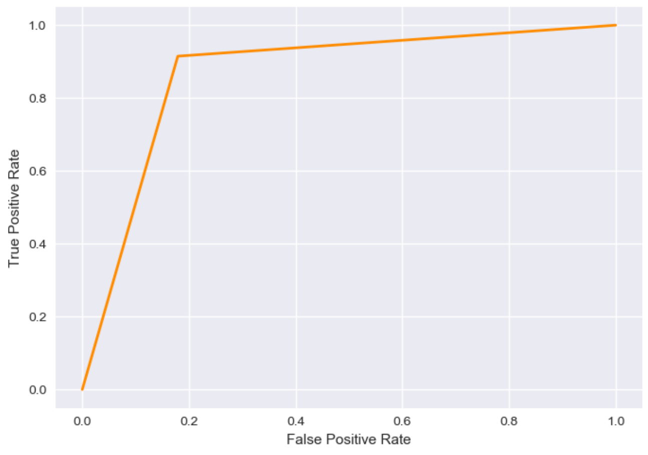


Figure 7.1: ROC Curve for SVM Model

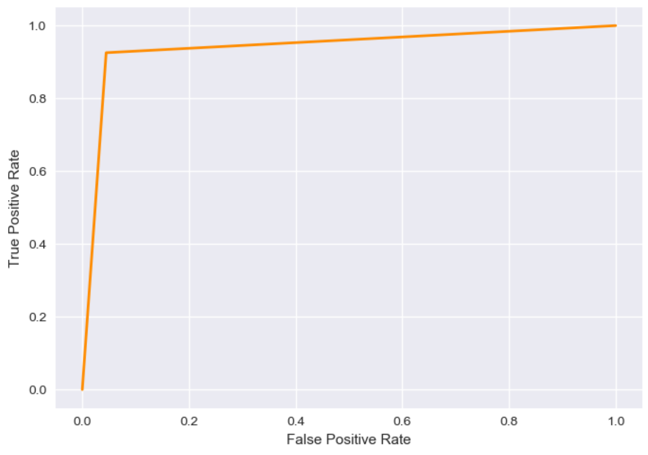


Figure 7.2: ROC Curve for Decision Tree Classifier

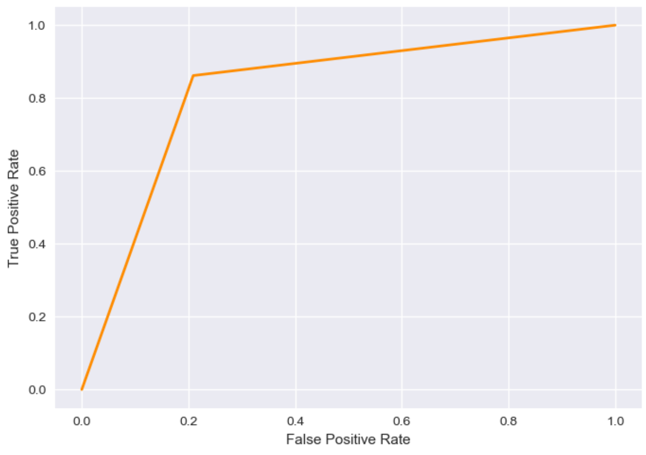


Figure 7.3: ROC Curve for Polynomial Linear Regression

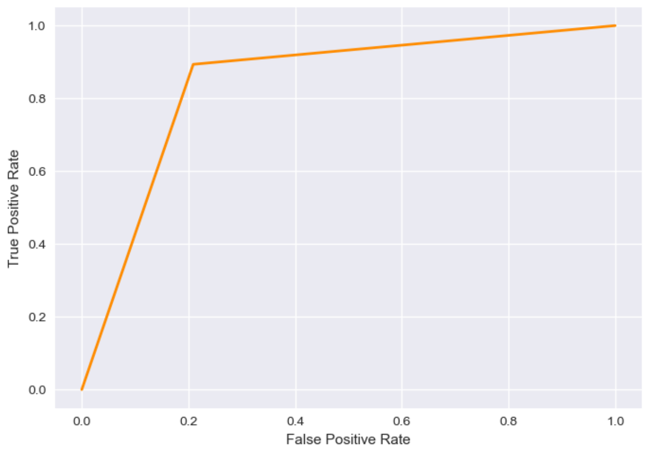


Figure 7.4: ROC Curve for Naive Bayes

Chapter 8

Discussion

In this section, we discuss the different ways we can use each module we have created in this work.

8.1 News Streaming Scraping Module

The first module we created in this work is the streaming and scraping module. This module allows us to extract news articles related to a topic from news websites. The module allows us to either extract articles previously published about the topic of interest, or to periodically check the website for updates about this topic. This way, this module can be used to extract articles from any online source about any topic of interest. This allows this module to be used in many problems other than the fake news detection problem. An example usage of this module is how we used it in this work: to extract articles about the major events that took place in the Syrian war from sources reporting the different points of views in this war.

8.2 Fake News Detection Model

The second module we created was the fake news detection model. We were able to build and train this model based on the Syrian war news. However, this model is able to label any news article or claim as fake or true regardless whether this article is related to the Syrian war or not. In order to use this model, we simply provide it with the content of the article. The model then tells us whether this article is true or fake.

8.3 Factcheck.org

Factcheck.org is a fact-checking website. In this section, we take a look at claims that were falsified or verfified by them why they said this article is fake or true what we

8.3.1 Example 1: Trump Claims New York Times Article Is Fake

In November 2016, a few days after Trump won the presidency, an article posted by the New York Times[1] quoting the Russian deputy foreign minister Sergei A. Ryabkov claimed that the Russian government maintained contact with Trump's "immediate entourage" during the presidential campaign. On the day that this article was posted, Trump tweeted that it is fake news.

False Statements on Russia - Factcheck.org

The article *False Statements on Russia* [2] was posted by the FactCheck website in the realm of George Papadopoulos, the foreign policy adviser to Trump's presidential campaign, pleading guilty to making false statements to FBI agents. He admitted that he lied about repeated contacts during the campaign with people he believed had ties to the Russian government.
Based on the above, along with other proofs, the fact-checkers say that **the article that Trump claimed was fake news was actually true**.

Running Our Model On The New York Times Article

We ran our model on the New York Times article in order to see if our model agrees with the fact-checkers. Based on our feature-extraction model, the article had low biased language, assertive, factive, implicative, hedges, and sectarian language frequencies, in addition to properly quoted sources.
Therefore, **our model labeled the article as true**.

Conclusion

In conclusion, our model agrees with the fact-checkers that **the article was actually true**, unlike Trump's accusations that it was fake.

[1]https://www.nytimes.com/2016/11/11/world/europe/trump-campaign-russia.html
[2]https://www.factcheck.org/2017/10/false-statements-russia/

8.3.2 Example 2: A Story Goes Viral About McCain Following His Death

In August 2018, a fake story about the Arizona Republican John McCain being responsible for an accident off the coast of Vietnam, which claimed 134 lives, injured another 161 and almost killed McCain went viral.

After McCain's Death, a False Claim Resurfaces - Factcheck.org

According to the research done by the fact-checkers [3], the official Navy investigation into the disaster showed the rocket actually misfired from the other side of the flight deck. Based on the records obtained from the Navy by a writer and editor who researched the event, the fact-checkers concluded that there is no possible way John McCain could have caused the fire on board the Forrestal.
Therefore, John MacCain was not responsible for the fatal 1967 accident, which means that **the articles reporting about this story are fake**. The fact-checkers given an example an article titled *"McCain The Hero Nearly Sunk an Aircraft Carrier and Killed 134 Sailors"* [4].

Running Our Model On An Article Reporting This Fake Story

We ran our model on the article titled: *"McCain The Hero Nearly Sunk an Aircraft Carrier and Killed 134 Sailors"* labeled by the fact-checkers to be fake.
Based on our feature-extraction model, the article had high biased language, assertive, factive, implicative, hedges, and sectarian language frequencies, in addition to not quoting proper sources in their claims.
Therefore, **our model labeled this article as fake**.

Conclusion

In conclusion, our model agrees with the fact-checkers that **the article was fake**.

[3]https://www.factcheck.org/2018/08/after-mccains-death-a-false-claim-resurfaces/

[4]https://www.theburningplatform.com/2018/08/28/mccain-the-hero-nearly-sunk-an-aircraft-carrier-killed-134-sailors/

Chapter 9

Conclusion

In this work we have presented a scraping and streaming model that allows users to scrape and stream any source for articles of interest and store them in the storage of choice such as HBase.
Using this model, we built a dataset of articles about the major events that took place in the Syrian war. We also made sure that this labeled dataset is not biased by automating the labeling step using fact-checking rather than relying on human bias and opinion.
Finally, we presented features relevant to the fake news problem that can be extracted from news articles and tested these features using machine learning models. The results show that our dataset and features performed well with all the tested machine learning models, with the lowest accuracy being 0.8 and the highest accuracy being 0.9.
In conclusion, we have managed in this work to create a feature extraction and machine learning model that are capable of extracting fake news-related features and detect if a given article is fake or not based on those extracted features with no need for a human factor to be involved.

Bibliography

[1] K. Shu, D. Mahudeswaran, S. Wang, D. Lee, and H. Liu, "Fakenewsnet: A data repository with news content, social context and dynamic information for studying fake news on social media," 09 2018.

[2] F. Torabi and M. Taboada, "The data challenge in misinformation detection : Source reputation vs . content veracity," 2018.

[3] J. Golbeck, M. Mauriello, B. Auxier, K. H. Bhanushali, C. Bonk, M. A. Bouzaghrane, C. Buntain, R. Chanduka, P. Cheakalos, J. B. Everett, W. Falak, C. Gieringer, J. Graney, K. M. Hoffman, L. Huth, Z. Ma, M. Jha, M. Khan, V. Kori, E. Lewis, G. Mirano, W. T. Mohn IV, S. Mussenden, T. M. Nelson, S. Mcwillie, A. Pant, P. Shetye, R. Shrestha, A. Steinheimer, A. Subramanian, and G. Visnansky, "Fake news vs satire: A dataset and analysis," in *Proceedings of the 10th ACM Conference on Web Science*, WebSci '18, (New York, NY, USA), pp. 17–21, ACM, 2018.

[4] W. Yang Wang, ""liar, liar pants on fire": A new benchmark dataset for fake news detection," 05 2017.

[5] N. Hassan, G. Zhang, F. Arslan, J. Caraballo, D. Jimenez, S. Gawsane, S. Hasan, M. Joseph, A. Kulkarni, A. K. Nayak, V. Sable, C. Li, and M. Tremayne, "Claimbuster: The first-ever end-to-end fact-checking system," *Proc. VLDB Endow.*, vol. 10, pp. 1945–1948, Aug. 2017.

[6] S. Mukherjee and G. Weikum, "Leveraging joint interactions for credibility analysis in news communities," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, (New York, NY, USA), pp. 353–362, ACM, 2015.

[7] K. Popat, S. Mukherjee, J. Strötgen, and G. Weikum, "Credibility assessment of textual claims on the web," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, (New York, NY, USA), pp. 2173–2178, ACM, 2016.

[8] W. A. Rugh, *Arab Mass Media: Newspapers, Radio, and Television in Arab Politics*. Greenwood Publishing Group, 2004.

[9] F. K. A. Salem and L. Tamim, "Communication balancing in the parallel gottfert algorithm," *Parallel Processing Letters*, vol. 22, no. 4, 2012.

[10] F. K. A. Salem, K. El-Harake, and K. Gemayel, "Factoring sparse bivariate polynomials using the priority queue," in *Proceedings of Computer Algebra in Scientific Computing*, vol. 8660 of *Lecture Notes in Computer Science*, pp. 388 – 402, Springer, 2014.

[11] F. K. A. Salem, K. El-Harake, and K. Gemayel, "Cache oblivious sparse polynomial factoring using the funnel heap," in *Proceedings of International Workshop on Parallel Symbolic Computation*, pp. 7–15, ACM Press, 2015.

[12] S. Ghemawat, H. Gobioff, and S. Leung, "The google file system," in *Proceedings of the 19th ACM Symposium on Operating Systems Principles 2003, SOSP 2003, Bolton Landing, NY, USA, October 19-22, 2003*, pp. 29–43, 2003.

[13] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber, "Bigtable: A distributed storage system for structured data," *ACM Trans. Comput. Syst.*, vol. 26, no. 2, 2008.

[14] J. R. Finkel, T. Grenager, and C. D. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling," in *ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA*, 2005.

[15] K. Knight, H. T. Ng, and K. Oflazer, eds., *ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA*, The Association for Computer Linguistics, 2005.

[16] K. Toutanova and C. D. Manning, "Enriching the knowledge sources used in a maximum entropy part-of-speech tagger," in *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora: Held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13*, EMNLP '00, (Stroudsburg, PA, USA), pp. 63–70, Association for Computational Linguistics, 2000.

[17] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *HLT-NAACL*, 2003.

[18] R. E. Khatib, J. E. Zini, D. Wrisley, M. Jaber, and S. Elbassuoni, "Topo-Text: Interactive Digital Mapping of Literary Text," in *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the*

Conference System Demonstrations, December 11-16, 2016, Osaka, Japan, p. (to appear), 2016.

[19] R. E. Khatib, J. E. Zini, D. Wrisley, S. Elbassuoni, and M. Jaber, "Topo-Text 2.0: Prototyping Modes of Interactive Mapping and Social Knowledge Creation," in *Innovative Interrogations: Modelling, Prototyping and Making, June 10-11, 2016, University of Victoria*, 2016.

[20] F. M. F. Wong, C. Tan, S. Sen, and M. Chiang, "Quantifying political leaning from tweets, retweets, and retweeters," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 8, pp. 2158–2172, 2016.

[21] F. M. F. Wong, C. W. Tan, S. Sen, and M. Chiang, "Quantifying political leaning from tweets and retweets," in *Proceedings of the Seventh International Conference on Weblogs and Social Media, ICWSM 2013, Cambridge, Massachusetts, USA, July 8-11, 2013.*, 2013.

[22] S. Volkova, G. Coppersmith, and B. V. Durme, "Inferring user political preferences from streaming communications," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pp. 186–196, 2014.

[23] C. Suen, S. Huang, C. Eksombatchai, R. Sosic, and J. Leskovec, "NIFTY: a system for large scale information flow tracking and clustering," in *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, pp. 1237–1248, 2013.

[24] S. Soni, T. Mitra, E. Gilbert, and J. Eisenstein, "Modeling factuality judgments in social media text," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 2: Short Papers*, pp. 415–420, 2014.

[25] Y. Sim, B. Acree, J. H. Gross, and N. A. Smith, "Measuring ideological proportions in political speeches," in *International Conference on Empirical Methods on Natural Language Processing 2013*, 2013.

[26] A. Prat and D. Strömberg, "The political economy of mass media," in *Advances in Economics and Econometrics: Tenth World Congress, 2013, Cambridge University Press*, 2013.

[27] V. Niculae, C. Suen, J. Zhang, C. Danescu-Niculescu-Mizil, and J. Leskovec, "QUOTUS: the structure of political media coverage as revealed by quoting patterns," *CoRR*, vol. abs/1504.01383, 2015.

[28] V. Nguyen, J. L. Boyd-Graber, and P. Resnik, “Lexical and hierarchical topic regression,” in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pp. 1106–1114, 2013.

[29] W. Lin, E. P. Xing, and A. G. Hauptmann, “A joint topic and perspective model for ideological discourse,” in *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML/PKDD 2008, Antwerp, Belgium, September 15-19, 2008, Proceedings, Part II*, pp. 17–32, 2008.

[30] Z. Jelveh, B. Kogut, and S. Naidu, “Detecting latent ideology in expert text: Evidence from academic papers in economics,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 1804–1809, 2014.

[31] M. Iyyer, P. Enns, J. L. Boyd-Graber, and P. Resnik, “Political ideology detection using recursive neural networks,” in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pp. 1113–1122, 2014.

[32] M. Ciot, M. Sonderegger, and D. Ruths, “Gender inference of twitter users in non-english contexts,” in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 1136–1145, 2013.

[33] B. A. Farraj, W. A. R. A. Orabi, C. Helwe, M. Jaber, O. Kayali, and M. Nassar, “Reconfigurable and adaptive spark applications,” in *Proceedings of the 32st Annual ACM Symposium on Applied Computing, 2017 (Submitted)*, 2017.

[34] A. Ghandour, M. Moukalled, and M. Jaber, “User-Based Load Balancer in HBase,” in *Proceedings of the 32st Annual ACM Symposium on Applied Computing, 2016 (submitted)*, 2017.

[35] B. Bonakdarpour, M. Bozga, M. Jaber, J. Quilbeuf, and J. Sifakis, “A framework for automated distributed implementation of component-based models,” *Distributed Computing*, vol. 25, no. 5, pp. 383–409, 2012.

[36] D. Wrisley, “Modeling the Transmission of al-Mubashshir Ibn Fatik’s Mukhtar al-Hikam in Medieval Europe: Some Initial Data-Driven Explorations,” *Journal of Religion Media and Digital Culture*, vol. 5, no. 1, pp. 228–257.

[37] W. Y. Wang, "“Liar, Liar Pants on Fire”: A New Benchmark Dataset for Fake News Detection," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pp. 422–426, 2017.

[38] R. Mihalcea and C. Strapparava, "The lie detector: Explorations in the automatic recognition of deceptive language," in *Proceedings of the ACLIJCNLP 2009 Conference*, pp. 309–312, 2009.

[39] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock, "Finding deceptive opinion spam by any stretch of the imagination," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, p. 309–319, 2011.

[40] S. Feng, R. Banerjee, and Y. Choi, "Syntactic stylometry for deception detection," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, p. 171–175, 2012.

[41] Z. Hai, P. Zhao, P. Cheng, P. Yang, X.-L. Li, G. Li, and A. Financial, "Deceptive review spam detection via exploiting task relatedness and unlabeled data," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pp. 1817–1826, 2016.

[42] V. Pérez-Rosas and R. Mihalcea, "Experiments in open domain deception detection," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, p. 1120–1125.

[43] F. M. Fouad, A. Sparrow, A. Tarakji, M. Alameddine, F. El-Jardali, A. P. Coutts, N. E. Arnaout, L. B. Karroum, M. Jawad, S. Roborgh, A. Abbara, F. Alhalabi, I. AlMasri, and S. Jabbour, "Health workers and the weaponisation of health care in Syria: a preliminary inquiry for The Lancet–American University of Beirut Commission on Syria," *The Lancet*, vol. 390, no. 10111, p. 2516–2526, 2017.

[44] D. Guha-Sapir, "Patterns of civilian and child deaths due to war-related violence in Syria: a comparative analysis from the Violation Documentation Center dataset, 2011–16," *The Lancet Global Health*, vol. 6, no. 1, 2018.

[45] H. Mowafi and J. Leaning, "Documenting deaths in the Syrian war," *The Lancet Global Health*, vol. 6, no. 1, 2018.

[46] A. Vlachos and S. Riedel, "Fact Checking: Task definition and dataset construction," in *Proceedings of the Workshop on Language Technologies and Computational Social Science at ACL 2014*, pp. 18–22, 2014.

[47] M. L. McHugh, "Interrater reliability: the kappa statistic," *Biochemia medica: Biochemia medica*, vol. 22, no. 3, pp. 276–282, 2012.

[48] Z. Huang, “Extensions to the k-means algorithm for clustering large data sets with categorical values,” *Data mining and knowledge discovery*, vol. 2, no. 3, pp. 283–304, 1998.

[49] P.-N. Tan, M. Steinbach, and V. Kumar, “Data mining cluster analysis: basic concepts and algorithms,” *Introduction to data mining*, 2013.

[50] S. Wold, K. Esbensen, and P. Geladi, “Principal component analysis,” *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.