Social Media vs. News Platforms: A Cross-analysis for Fake News Detection Using Web Scraping and NLP

Fahad Alsuliman
Florida Institute of Technology
Melbourne, Florida, USA
falsuliman2019@my.fit.edu

Siddhartha Bhattacharyya
Florida Institute of Technology
Melbourne, Florida, USA
sbhattacharyya@fit.edu

Khaled Slhoub
Florida Institute of Technology
Melbourne, Florida, USA
kslhoub@fit.edu

Nasheen Nur
Florida Institute of Technology
Melbourne, Florida, USA
nurn@fit.edu

Candice Normalee Chambers
Florida Institute of Technology
Melbourne, Florida, USA
chambersc2017@my.fit.edu

ABSTRACT

With the widespread use of social media platforms within our modern society, these platforms have become a popular medium for disseminating news across the globe. While some of these platforms are considered reliable sources for sharing news, others publicize the information without much validation. The transmission of fake news on social media impacts people's behavior and negatively influences people's decisions. During the COVID-19 outbreak, it was more evident than ever. This has led to a demand for conducting research studies to explore sophisticated approaches to assess the integrity of news worldwide. The main objective of this research paper was to outline our proposed experimental methodology to detect and access fake news using Data Mining and Natural Language Processing. The presented research effort provides a method to verify the authenticity of the news disseminated in social networks by dividing the process into four significant stages: news aggregation, publication collection, data analysis, and matching results.

CCS CONCEPTS

• **Human-centered computing → Collaborative and social computing theory, concepts and paradigms**; • **Computing methodologies → Information extraction**; • **Applied computing → Sociology**; **Evidence collection, storage and analysis**.

KEYWORDS

Social Media, Web Crawling, Fake News Detection, Web Scraping, Natural Language Processing

ACM Reference Format:
Fahad Alsuliman, Siddhartha Bhattacharyya, Khaled Slhoub, Nasheen Nur, and Candice Normalee Chambers. 2022. Social Media vs. News Platforms: A Cross-analysis for Fake News Detection Using Web Scraping and NLP. In *The15th International Conference on PErvasive Technologies Related to Assistive Environments (PETRA '22), June 29-July 1, 2022, Corfu, Greece.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3529190.3534755

1 INTRODUCTION

Nowadays, the ease of publishing and disseminating news on social networks has helped spread both correct and false information among the community. A survey by Statista showed that 49% of Americans intentionally or unintentionally share fake news [17]. In addition, this false information has adverse effects on health, politics, and social issues. The false rumor on social media that COVID-19 is linked to 5G technology was widely circulated and believed, and many people acted on it. A study in [10] showed that 34.8% of Twitter users think there is a connection between 5G and COVID-19. Additionally, a study has shown that the implementation of bots spreading fake news during the 2016 U.S. Presidential Elections influenced voters based on incorrect information [4]. An example of the economic impact that resulted from spreading rumors on social networks was demonstrated in 2019, when a Twitter account posted a fake video of a Tesla crashing into a robot prototype, causing the company's shares to plunge by $20 [11].

As part of this study, the research corroborates the legitimacy of the news spread on social media to enable users to verify the content presented to them. The flexibility of searching on social media offers users many different options, but it does not provide information about the legitimacy of the sources. This research delivers a methodology that collects publications from social media networks, collects news from reliable news sources, and contrasts them against each other by applying machine learning and Natural Language Processing techniques.

Our Contributions: We developed a pipeline to cross-examine social media posts with reliable news sources to detect false information. Contributions to this paper are threefold. Firstly, our news sources (Fox News and CNN) lean toward different political views. Using these two trusted news sources helps increase the chances of detecting fake news and removes biases toward the specific political view. Tweets from the news sources were not used, as all news articles did not have a Tweet; also, news sources' tweets are short (280 characters) as a result, they did not serve as a promising matching source. It was found that longer articles gave more accuracy. Secondly, we did not depend on a pre-specified dataset for our queries. We used our social media scraper to select random posts to minimize human involvement and project assessment. Some of

PETRA '22, June 29-July 1, 2022, Corfu, Greece
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9631-8/22/06…$15.00
https://doi.org/10.1145/3529190.3534755

these posts were written in a simple language and had slang. Our algorithm was able to compare it against news articles that use formal language and retrieve related articles. Lastly, the soft inverse document frequency was used instead of the regular inverse. IDF is usually adopted in many research papers, whereas the soft IDF has proven to be better in comparing a short text against a long one[15].

2 LITERATURE SURVEY

Ferreira and Vlachos [7] conducted a study on the Emergent project's data with 2595 news articles and 300 claims linked to these publications. They used the data set to perform stance detection using NLP for fact-checking and developed the stance classification model based on MLR (multi-class logistic regression). Although they conducted their investigation on a small dataset that contained insufficient information for efficient distinction, they achieved a 73% accuracy [7]. Many research focuses on the political domain; however, this concentration limits the algorithm's performance when used on different domains. Another research study [1] utilized a machine learning ensemble approach to detect fake news. The data used for the experiment were collected from other domains of public web pages accessible through the Internet to distinguish fake news from real news. Textual features were extracted using LIWC and used as models' input. The logistic regression model showed an accuracy of 90%, and LSVM scored an accuracy of 96%.

Pranav Ashtaputre et al.[9] made a model based on Web scraping data on reddit.com to create a dataset that divides news into legitimate and fake categories. The dataset is processed using NLP and split into train and test datasets. Four models were generated using the pipeline; combined Logistic regression and TFIDF Vectorizer resulted in the best accuracy of 86.25 %. Hannah Rashkin et al. [12] conducted a study focusing on news media language for fake news detection. The study compared the language of fake news against the language of real news to determine the truthfulness of the articles. They collected 20k articles from various news sources as training data. The LSTM model took the text as input and predicted truth ratings; its accuracy score was 56%, revealing the classification challenges in Fake New Analysis but highlighting lexical features' usefulness.

Alternatively, other researchers focused more on processing data with NLP than the data collection or scraping part. Shuy et al.[14] used a data mining approach while web scraping that focuses on social media articles to detect fake news. The central concept focuses on the News and Social Context Features; they extracted features using Linguistic analysis centered on text (source, headline, body text) and Visual extraction on videos and pictures. These mined data construct the model based on either News Content (knowledge-based or style-based) or Social Context (stance-based or propagation-based) that decides whether the news articles evaluated are fake or not. Asad et al. and Erascu et al.[2] concentrated their experiment on a Click-bait and Non-clickbait dataset containing 32000 news titles. Parsing is a method of extracting content from a website based on HTML code. At the same time, machine learning uses API and cosine similarity to focus on categorizing the title as clickbait or not. The LSVC and TF-IDF model scored a 95.6% accuracy, while the bigram approach achieved 84.5%. Web Scraping is a valuable process that automates extracting information from the World Wide Web. Han et al. explained that this feature could be proven cost-efficient as opposed to manually filtering out large sums of data [8]. Boegershausen et al. point out that as much as Web Scraping techniques are resourceful, the information extracted is not validated and poses challenges when inquiring about its legitimacy [5].

3 OVERALL WORKFLOW OF THE PROPOSED METHODOLOGY

In this research effort, News sites are classified as trusted sources. Typically, reliable sites have many reviewers and high credibility in their community. After selecting these sites, the crawler crawls the news links of the selected news sites for a certain period. Then, it utilizes sitemaps, RSS, and official Twitter accounts as sources for news links. Links are then stored in a database along with the respective news headline. The data includes the article's body, title, and date. The designed tool sends a request to each collected link and extracts news articles using scraping methods. Then, the posts are aggregated from social media using 3rd party libraries. The data is refined from the additional tags. Natural Language Processing (NLP) techniques are executed to prepare the data for processing. The data is passed to the processing component, converting text to numerical representation using TF-IDF values. Three methods are applied to match the data and predict authenticity: cosine similarity, matching score, and word appearance. Fig. 1 shows the overall workflow of our approach.

The overall workflow of our methodology depends on dividing the system into micro-services. These services integrate and work together as one unit. The first part is the **crawler**. The crawler's goal is to find and return articles' links that meet the requirement. The crawler always has a starting point that is represented as a specific page URL. In this application, two different approaches to crawling are used, which are further discussed in the proposed methodology section. The crawler feeds the page content to the **data cleaner** component. The data cleaner's goal was to eliminate all undesirable links from the page content. For example, removing links to contact info, copyright text, and navigation links. Also, the data cleaner does not return unrelated links, as keywords govern the cleaner. Finally, the data cleaner stores all the retrieved links in the database.

The **database** serves as a central hub for all components of the systems; for this research, MySQL database was selected. The retrieved links, articles, providers' info, and queries were all stored in the database. The database tables were linked using primary and foreign keys to enhance performance. The next component, **scraper** retrieved the links crawled by the crawler from the database. It sends a request to each link to extract the page content. Each news agency website had a different scraper based on the technology used to build their website. The scraper then sends the pages' content to the **scraper's data cleaner**. The data cleaner outputs only the required text without unnecessary information; the author's names, ads, links, symbols, etc., were removed from the page content. The scraped articles were then stored in the database.

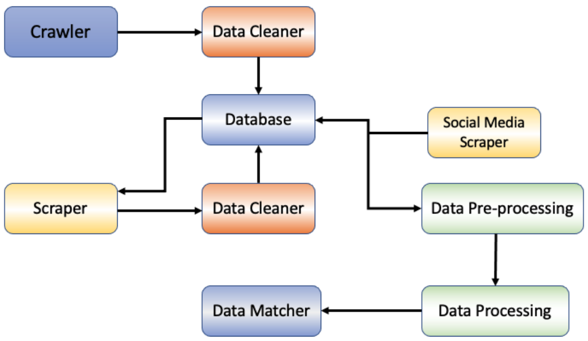

Figure 1: Overall Workflow of the Proposed methodology

Social Media Scraping: The goal of this component was to retrieve COVID-19 related posts and store them in the local database. This research focused on two major social media networks, Twitter.com and Facebook.com. The facebook-scraper is used for Facebook and Twint- a python tool for Twitter. Configuration for these two tools is discussed in detail in the proposed methodology section.

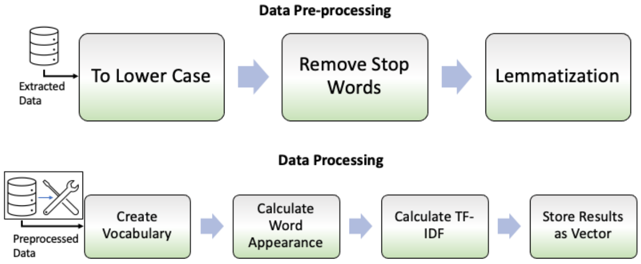

Figure 2: Data Pre-processing and Processing Steps

Data Pre-processing: The data pre-processing phase makes the extracted data from social media and global news network sites ready for processing. This step includes retrieving texts from databases, dividing articles into single words and processing them separately, then processing the entire article and preparing it for the next step. The first example of single-word processing is converting numbers to written numbers, as written numbers can be converted into vector values in the following steps. The pre-processing phase includes deleting stop words, converting text to lowercase, and lemmatizing, further discussed in the next section.

Data Processing: This step aimed to convert pre-processed text from articles and social media publications into numbers represented as vectors that can be processed and compared using natural language processing tools. This step involved creating a dictionary that holds all the articles' words. The dictionary counts the word counts throughout all documents and appears in a single article. This step also supports the computation of the TF-IDF value and assigns a weight for each word. Fig. 2 shows the data pre-processing and processing steps.

Data Matching: After converting data into numbers in the form of vectors, social media posts are compared with news articles to determine the posts' authenticity and their closeness to the truth. Three methods were used to compare data. The first method used was the cosine similarity which checks the angle between two vectors. The second method is called matching score, where the weights of words are added that appeared in both the article and the query. The last method checks the number of words in the article and the query.

4 APPLYING THE PROPOSED METHODOLOGY

Web Crawling and Collecting Links : In this research, CNN and FOXNews websites were used as sources of verified and trusted news articles. To extract the list of articles' links, the sitemaps of each website have been used, which are built using different technology. The sitemap is based on HTML and CSS tags for CNN, while FOXNews uses XML tags. For this purpose, two different classes are used to crawl and extract links from each website. The health article links are stored on a single page grouped by month for CNN. The application retrieves the list of links by looking for the CSS class(li). Then a data extraction method is applied to retrieve URLs from spans with a class name:sitemap-link. Articles titles were extracted from the same span. Links, titles, and provider IDs were then stored in a local MySQL database using MySQL.connector.connection class. The flag of non-scraped articles was designated to be 'N.' The flag was used to identify scraped/non-scraped articles.

The above approach depends on CSS tags, so it does not apply to FOXNews.com sitemaps. For this purpose, the Beautiful soup library[1] was used. Beautiful soup is an analytics engine based on HTML/XML used to analyze and pull data and information in the DOM tree out of HTML and XML files[19]. A request was sent to the FOXNew.com sitemap using the urllib[2] library. That request was passed to Beautiful soup to extract the data using soup.findall('loc') where loc represented each row in the XML document. After extracting all links, the links were further filtered by searching for health-related news articles using the search keyword 'health.' These links and their titles were stored in the database to be scraped for content.

Web Scraping: Web scraping is extracting contents from the World Wide Web and storing them in a system file or a database for later analysis [18]. CNNScraper class retrieves all CNN articles links (provider id = 1) that are not scraped yet (scraped = N) from the database. Links will be stored in a python list. The application will iterate over the list. The app will send a get request for each link to retrieve the page content. The page content will be passed to the beautiful soup library with html.parser.soup = BeautifulSoup(page.content, "html.parser"). The scraper will first extract the article date from the URL. Then will retrieve data from the paragraph titled "zn-body__paragraph speakable," representing the first part of the article body. Then, the rest of the article content will be

[1] https://www.crummy.com/software/BeautifulSoup/bs4/doc/
[2] https://docs.python.org/3/library/urllib.html

extracted from div class "zn-body__paragraph". The two parts will then be merged and converted from a list to text. All HTML tags will be removed using

$$re.findall(r' > (.*?) <', str(body\_first\_part)).$$

The extracted text is passed to a method to decide if the article is related to COVID-19 news. The related Articles method has a set of words including ("COVID", "COVID19", "COVID-19", "CORONA", "Coronavirus", "Novel strain"..etc) if the article body has one of these words, the article is stored in the database for future analysis and the scraped flag updated to 'Y .'Otherwise, it is marked as an unrelated article.

FOXNews.com is not as organized around tags as CNN.com. Therefore, all paragraphs were extracted from the targeted page. Then all extra information were removed using the data cleaner class. Removed data includes contributions, author bio, image descriptions, image copyright, and ads.

Data Cleaning for News Articles: To prepare the data for processing, data cleaning techniques have been applied that increase the utility of the data. Two separate data cleaners were designed in this project, social media posts data cleaner and news articles data cleaner. The data cleaner performed different actions to ensure the data was ready for the upcoming steps:

- It removed any HTML tags, as these tags could affect the data processing pipeline. It also removed any special characters that are not meaningful to the article body. After that, it checked for extra spaces and removed unnecessary white spaces from the article body.
- It removed all unrelated text that crept into the body, for example, contribution, author bio, and image description.
- It removed all punctuation that had no meaning in machine language.

Social Media Scrapers: Facebook and Twitter websites have their scraper based on the technologies used on the platform. The best way to scrape social media websites was identified to be the official API. Unfortunately, both APIs don't meet the specifications for this project. Twitter API (free version) only returns results for the last seven days. At the same time, Facebook API needs a contract and proof of business to provide the user access to their API. For these purposes, external libraries were used to scrape these platforms.

For Twitter.com, the Twint[3] library was used, which is an advanced python tool to scrape Twitter without API. Utilizing Twitter's search operators, Twint allows scraping Tweets related to specific topics, hashtags, and trends and sorting out sensitive information such as e-mail addresses from Tweets. Twint must be configured to work and retrieve the correct information. The configuration includes the search dates, output type, Panda integration, the limit, etc. To meet the objectives of this research, three generic methods were created to leverage the Twint library: searchByUserAndTweet ,searchTweetByText, and searchTweetByUser. The data range was set for retrieved tweets to be between 07/01/2021 to 08/31/2021; this date represents the range of articles scraped. After collecting the tweets, four cleaning methods methods were applied: delete_http_links, delete_usernames, delete_html_tags, and delete_punctuation. These methods are part of the data cleaning step and aim to prepare the data for the preprocessing stage. For Facebook.com, facebook_scraper[4] was used. The tool retrieves data from public pages. The data include posts, comments, and engagement. The get post method takes four arguments: page name, user name, password, and how many pages to scrape. All posts are accumulated in a Pandas data frame to apply the cleaning methods and then stored in the database.

Data Pre-processing Stage: The data pre-processing step is preparing the data to be converted to vectors to improve the results of text retrieval. This phase is part of the NLP process. It includes different steps such as removing stop words, converting the text to lower case, converting numbers to text, and Lemmatization. The first step involved converting the whole text to lower case to maintain consistency. After that, stop words were removed. Stop words such as "the, a, etc." hold no value and are counted as useless words. Usually, these words take space in the database and slow the NLP process. The NLTK library (Natural Language Toolkit) was used to remove these words. First, the article was converted to tokens using NLTK tokenizer, and then each word was checked against the NLTK stop word. The stop word was removed from the processed texts. The next step was to convert numbers to their textual representation. For example, '1 was converted to 'one.' The num2words library was used for this step. Num2words is an open-source python library that converts numbers to words and supports multiple languages. To ensure that num2words does not introduce any new stop words, the remove stop words step is executed again.

Next, each word was lemmatized in the text. According to [3], "lemmatization removes inflectional endings and returns the base or dictionary form of a word." Lemmatization is usually used with text retrieval techniques and search engines. With Lemmatization, more accuracy of text retrieval operations can be achieved. For this step, WordNetLemmatizer from NLTK was used. The library uses WordNet, an English word lexical database, to perform the lemmatization operation. Lemmatization outperformed stemming in Balakrishnan's experiment[3]; so Lemmatization was selected for this project.

Data Processing : After pre-processing the articles list and the query, the resulting texts are sent to the data processing component. The goal of this component is to convert the text to numerical representation, which can be used to compare texts using different methods. TF-IDF was used (term frequency-inverse document frequency) to perform this step. Term frequency gives a weight to each term in a document depending on the number of appearances of the word in the document[13].

$$TF(t,d) = \frac{Term\ t\ frequency\ in\ document\ d}{Total\ words\ in\ document\ d}$$

With TF only, the limitation is that the same importance level is given for similar terms across the corpus. For example, it is highly likely in our data set to have the word "COVID" appear in every document. Documents will have different contexts, "COVID" might appear in a document related to the Olympics games article. So, inverse document frequency was used to solve the issue. The IDF of a term that rarely appears will be high, giving more accuracy

[3] https://github.com/twintproject/twint

[4] https://github.com/kevinzg/facebook-scraper

to data retrieval operations. IDF focuses on how many documents have the term. For this experiment, the smoothed IDF was used as it appears in [6]. The IDF formula is as follows:

$$IDF_t = Log\frac{Total\ documents}{Total\ documents\ with\ term\ t}$$

$$IDF_t = Log\frac{Total\ documents}{Total\ documents\ with\ term\ t\ +1}$$

TF was multiplied by IDF to get the weight for each word.

$$TF - IDF_{t\_,d} = TF_{t\_,d} * IDF_t$$

Firstly, the vocabulary to calculate the TF-IDF score was created. In order to do so, all pre-processed articles were retrieved and tokenized. Then, each word was added to the dictionary along with the assignment of a unique ID. After that, the number of documents that had the term was counted and stored in a separate data set. The TF-IDF can then be computed for each word in each document using these two data sets. The result was stored in a 2D array, where rows represented documents and columns represented words. If a word appeared in the document, it had a TF-IDF score in the row of the document; otherwise, the value was a 0.

Data Matching: After processing the data, the resulting vectors are passed to three different data matches—cosine similarity, matching score, and word appearance. The goal of the data matcher is to calculate the similarity between two texts (article and social media post).

(1) **Cosine similarity** calculates the cosine angle between two vectors. The cosine similarity method determines the similarity based on orientation, not magnitude. The smaller the angles, the more similar the two texts are. According to [16], cosine similarity with TF-IDF outperformed cosine similarity with Word2Vec vectors.

$$\cos(\theta) = \frac{A \cdot B}{||A|| \cdot ||B||}$$

(2) **Matching score**: This step added the TF-IDF values for the common words between article and query. Finally, this method returned the article with the highest TF-IDF sum.

(3) **Word appearance**: This method depended on calculating the number of words that appeared in both the article and the query. Finally, this method returned the article with the most number of words in common with the query.

5 RESULTS

In the experiments, 521 news articles were scraped from CNN and FoxNews. The articles were published between 07-01-2021 and 08-31-2021. In comparison to suspected fake news, these articles were considered reliable sources of information. Ten tweets from Twitter were used as queries. All tweets were randomly selected with different topics related to COVID-19 and the US. A Twitter search was conducted using ten different phrases to collect the tweets, and it was chosen randomly from the results extracted by the Twitter scraper.

A greedy approach was followed while selecting the best threshold for each similarity score. In this process, initially, a random number was selected for the threshold, then the accuracy of predictions was checked by looking into the outcome; this process was continued until the chosen threshold showed the best results. For cosine similarity and matching score, a threshold of 0.15 was considered to identify if the news was fake or did not exist in the created dataset. If the most relevant article scored under 0.15, the query was regarded as fake or didn't exist. For word appearance, 60% was the chosen threshold. If the top article had less than 60% of query words after preprocessing, the query would be considered fake or doesn't exist. The tweet topics considered were new regulations, 5g theory, COVID symptoms, spread mechanism, death rumors, and Florida COVID news. Tweets that didn't have information were removed and replaced by another random tweet.

Table 1 shows the similarity score and the truth values (F= Fake , R= real) based on the threshold(0.15) between the query and the most similar article against the truthfulness of the query inspected manually. Some examples of Fake news that were correctly identified are queries 4 and 5. Query 4 had the text "Covid and 5G are linked. You won't change my mind". Query 5 had the text "Actor Mike Mitchell Dies of Heart Attack 7 Days After 3rd Pfizer COVID Vaccine Booster".

Table 1: Similarity Scores for Ten queries

Queries	Cosine similarity	Matching Score	Word App	Ground Truth: Fake or Real
Q1	0.108,F	0.118,F	0.645,R	R
Q2	0.187,R	0.235,R	0.520,F	R
Q3	0.242,R	0.276,R	0.833,R	R
Q4	0.087,F	0.040,F	0.571,F	F
Q5	0.101,F	0.132,F	0.583,F	F
Q6	0.237,R	0.218,R	1,R	R
Q7	0.185,R	0.264,R	0.666,R	F
Q8	0.220,R	0.311,R	0.944,R	F
Q9	0.285,R	0.435,R	0.333,F	F
Q10	0.176,R	0.159,R	0.833,R	R

Word appearance method performed better with a 70% accuracy in classifying fake news whereas, cosine similarity and matching score had a 60% accuracy. However, after reviewing the retrieved articles (with the top scores), it was observed that it was based on unrelated articles, which made the method unreliable. To decide, if a method performed well, it was check if the retrieved article was related to the query or not. This was done by inspecting each article manually to decide the accuracy - Y = related , N = not related (See Table 2).

Table 3 shows the scoring method for the similarity scores with the relatedness of the articles. Cosine similarity achieved a 80% accuracy, where matching score and word app scored 70% and 40% respectively.

6 DISCUSSION

The world is in a state of uncertainty due to the Coronavirus spreading worldwide. The average person is looking for news updates

Table 2: Relatedness of the queries

	Cosine similarity	Matching Score	Word App
Q1	Y	Y	N
Q2	N	N	N
Q3	Y	Y	Y
Q4	N	N	N
Q5	Y	N	N
Q6	Y	Y	Y
Q7	Y	Y	N
Q8	Y	N	N
Q9	N	N	N
Q10	Y	Y	Y

Table 3: Method performance in helping detect fake news based on the relatedness of articles

Prediction	Actual result	Provide a related article	Score
False	False	Yes	1
False	False	No	1
False	True	No	0
False	True	Yes	1
True	True	No	0
True	True	Yes	1
True	False	No	0
True	False	Yes	1

more intensely than before. Social media such as Facebook and Twitter were the closest and fastest way to disseminate and broadcast news. Unfortunately, some people or entities have used these means to spread false information, intentionally or unintentionally. In some cases, this news caused panic in individuals. In this study, a methodology was developed to investigate the authenticity of social media news without human intervention. The approach relied on two of the most prominent American news agencies (FoxNew.com and CNN.com) as verified news sources. The data was extracted using web scraping and web crawling. The extracted data from social media platforms were then cleaned by removing the author's name, copyrights, and text parts that do not have value, such as HTML symbols. Next, NLP was applied to this data by returning words to their roots (Lemmatization, removing special characters) and converting numbers to letters.

The preprocessed data was then converted into a numerical representation. In this step, the TF-IDF value was computed for every word in each article and post from the social network. In the experiments, three methods were used to compare numerical data and calculate their similarity. The first method was cosine similarity, representing the angle between two vectors, as each vector represents a specific text. The smaller the angles, the closer the two texts. The TF-IDF values were summed for words contained in both the article and the query on the other approach. Then the article with the highest sum was retrieved. The common words were counted between the article and the post in the third method.

In the experiments, two months of data were extracted that contained 521 articles from the mentioned news agencies and ten randomly selected tweets from Twitter.com. The counting method had the highest score for classifying fake news with a 70% accuracy, while 60% for both cosine similarity and matching score. However, after reviewing the news that helped make the prediction, it was found that the prediction in the word counting method was biased towards information unrelated to the post. The accuracy of retrieving a related article was only 40%. In contrast, cosine similarity and the matching score had 80% and 70% accuracy in retrieving related articles. Therefore, it was concluded that counting words were ineffective. The cosine similarity method identifies false news most effectively since it provides information related to the post that helps debunk false information. It is hypothesized that increasing the size of the dataset will result in higher accuracy by including more reliable sources of information.

7 CONCLUSION AND FUTURE WORK

In conclusion, identifying false news is a significant challenge. The presence of simple words such as negative words changes the context of the post but still scores high compared to a similar but opposite article. For example, when the word "not" is added to "Florida's COVID cases are rising, " it becomes " Florida's COVID cases are not rising, " which gives an entirely different meaning. The comparison method will return an incorrect prediction based on the significant similarity with one of the articles. Still, at the same time, it will return an article that helps debunk this fake news.

Our future plan is to extend the project by adding articles with different topics to the dataset. We plan to expand the datasets to other social media platforms and other topics and domains. Moreover, we want to compare biased and unbiased topics and public opinions in the news media to evaluate the efficiency of our approach. For example, both CNN and Fox News may provide data supporting and opposing events such as wars and political leadership. Therefore, testing our method on biased and unbiased media sources and topics will be a part of longitudinal research. Moreover, increasing the size of the training datasets may improve prediction accuracy. These data sets could be used to build a machine learning model that could be deployed as a browser extension. In addition to suggesting the possibility of fake news to the user, the extension could provide a link that helps clarify the information.

REFERENCES

[1] Iftikhar Ahmad, Muhammad Yousaf, Suhail Yousaf, and Muhammad Ovais Ahmad. 2020. Fake news detection using machine learning ensemble methods. *Complexity* 2020 (2020).

[2] Bashar Al Asaad and Madalina Erascu. 2018. A tool for fake news detection. In *2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*. IEEE, 379–386.

[3] Vimala Balakrishnan and Ethel Lloyd-Yemoh. 2014. Stemming and Lemmatization: A Comparison of Retrieval Performances. *SCEI Seoul Conferences* (2014).

[4] Alessandro Bessi and Emilio Ferrara. 2016. Social bots distort the 2016 US Presidential election online discussion. *First monday* 21, 11-7 (2016).

[5] Johannes Boegershausen, Abhishek Borah, Hannes Datta, and Andrew Stephen. 2021. Fields of Gold: Generating Relevant and Credible Insights Via Web Scraping and APIs. *ACR North American Advances* (2021).

[6] Cedric De Boom, Steven Van Canneyt, Thomas Demeester, and Bart Dhoedt. 2016. Representation learning for very short texts using weighted word embedding aggregation. *Pattern Recognition Letters* 80 (2016), 150–156.

[7] William Ferreira and Andreas Vlachos. 2016. Emergent: a novel data-set for stance classification. In *Proceedings of the 2016 conference of the North American chapter*

of the association for computational linguistics: Human language technologies. ACL.

[8] Saram Han and Christopher K Anderson. 2021. Web scraping for hospitality research: Overview, opportunities, and implications. *Cornell Hospitality Quarterly* 62, 1 (2021), 89–104.

[9] Mayank Kumar Jain, Dinesh Gopalani, Yogesh Kumar Meena, and Rajesh Kumar. 2020. Machine Learning based Fake News Detection using linguistic features and word vector features. In *2020 IEEE 7th Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*. IEEE, 1–6.

[10] Wasim Ahmed Joseph Downing. 2020. COVID-19 and the 5G Conspiracy Theory: Social Network Analysis of Twitter Data. *J Med Internet Res* (2020).

[11] David D Parsons. 2020. The impact of fake news on company value: evidence from tesla and galena biopharma. *CHANCELLOR'S HONORS PROGRAM PROJECTS* (2020).

[12] Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. 2017. Truth of varying shades: Analyzing language in fake news and political fact-checking. In *Proceedings of the 2017 conference on empirical methods in natural language processing*. 2931–2937.

[13] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2007. An introduction to information retrieval.

[14] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *ACM SIGKDD explorations newsletter* 19, 1 (2017), 22–36.

[15] Pinky Sitikhu, Kritish Pahi, Pujan Thapa, and Subarna Shakya. 2019. A comparison of semantic similarity methods for maximum human interpretability. In *2019 artificial intelligence for transforming business and society (AITB)*, Vol. 1. IEEE, 1–4.

[16] Pinky Sitikhu, Kritish Pahi, Pujan Thapa, and Subarna Shakya. 2019. A comparison of semantic similarity methods for maximum human interpretability. In *2019 artificial intelligence for transforming business and society (AITB)*, Vol. 1. IEEE, 1–4.

[17] Amy Watson. 2020. Statista: Sharing of made-up news online in the U.S. 2019. https://www.statista.com/statistics/657111/fake-news-sharing-online/

[18] Bo Zhao. 2017. Web scraping. *Encyclopedia of big data* (2017), 1–3.

[19] Chunmei Zheng, Guomei He, and Zuojie Peng. 2015. A Study of Web Information Extraction Technology Based on Beautiful Soup. *J. Comput.* 10, 6 (2015), 381–387.