Navigating News Bias: A Visual Perspective

Andrew Zhang, Bryan Zhao, Neel Ganediwal, Yash Gupta, Aayush Seth, Shiva Ramaswami

Team #15

1 Introduction

With the presidential election right around the corner and the political landscape becoming ever more divided, is it more important than ever for people to understand and identify bias in media. Many news and social media sites are incentivized to increase user retention, and the resulting clickbait headlines and potentially hyper-politicized news articles form echo chambers which have been shown to increase polarization [10]. Identifying media bias and providing access to a variety of viewpoints helps combat misinformation and allows readers to be better informed. This can help improve public sentiment on current events, since well-informed readers with knowledge of a variety of viewpoints tend to have less anxiety and fear about said current events [2].

2 Problem Definition

We want to create a system to not only detect bias in news articles, but to also interactively convey the that bias to users. We aim to tackle two main computation tasks. First, we want to identify different types of bias within news articles using various NLP and classification methods. Second, we want to group articles by topic using keyword extraction and clustering. Combined, this allows us to create an interactive recommendation system where users can view articles with similar topics across a large range of biases, promoting a wider variety of viewpoints. Moreover, grouping articles by topic allows the system to display the distribution of biases across articles on a particular topic, offering users insights on biased media coverage and helping them distinguish between biased and nonpartisan content.

3 Literature Survey

There have been many approaches to detecting bias in documents using machine learning and NLP techniques. Work has been done to investigate gender bias in word embeddings [16], where certain words and phrases are inherently associated with certain genders. While potentially useful in detecting political bias, this is likely too fine-grained to be used for larger scale news articles with many words. Other works have utilized various machine learning models such like SVM [6] or deep learning models such as RNN [4], LSTM [3] [9], BERT [3] [19], and Attention Networks [9] to classify and predict bias in articles. While they provide insight for classifying bias with their approaches to collecting and processing data as well as creating and fine-tuning models, they either focus on simply detecting the presence of bias or ternary labels of left, right and center. These works lack investigation into predicting more fine-grained bias labels that can provide more nuanced classifications. To develop our methods, we are additionally investigating the use of language model embeddings such as BERT [5] and DistilBERT [18] which have been shown to perform well on a wide variety of downstream NLP tasks, including text classification.

Work has also been done to investigate bias using named entity search [14], determining how different news sources disproportionately cover different topics and/or political parties. While the authors present a new form of bias in news called coverage bias, the experiment investigated a limited number of news sources and failed to provide insight on determining bias on a per-article level. There have also been efforts to map news outlets along a political spectrum to create political media maps [1]. While the categorization was more fine-grained, it focuses on news outlets, rather than individual news articles and their content.

Apart from bias detection, many works also investigate keyword extraction on documents or a text corpus, using various machine learning techniques. This includes various statistical and graph-based models for unsupervised keyword extraction including PositionRank [8], TextRank [15], and RAKE [17]. The strengths and weaknesses of each are compared in [22], which provides crucial insight on the usage of the aforementioned methods, noting that PositionRank seems to provide the best results. More recently, a new pre-training objective, Keyphrase Boundary Infilling with Replacement [13], was introduced that significantly improves the performance of transformer language models for keyword extraction. While all of these methods provide avenues for extracting keywords, they operate on a per-article basis, aimed at summarizing a body of text, rather than grouping articles of similar keywords.

Finally, on conveying and visualizing the presence of bias in articles, [21] evaluates different methods of conveying the presence of bias in articles, and the authors find that presenting simple labels indicating bias is ineffective in alerting readers to existing bias. Additionally, [20] finds that simply displaying opposing articles side-by-side is also relatively ineffective in promoting users to investigate opposing viewpoints. [11] demonstrated a matrix-format of visualizing opposing articles from different counties, but requires modification to visualize bias on a certain topic. These studies highlight the need for a interactive tool to recommend articles of opposing viewpoints, and provide baseline visualization formats that can be expanded upon.

4 Our Approach and Innovations

Our bias visualization system presents three core innovations. First, our bias classification model aims to detect more fine-grained bias labels on a scale from 0 to 4 instead of the normal binary or ternary labels that only indicate the presence of bias. This will allow for more nuanced analysis, allowing users to not only identify opposing viewpoints, but also to take note of and avoid hyper-partisan content. Second, we build on typical keyword extraction methods by additionally performing keyword clustering. This allows our system to additionally group articles with similar topics based on these keywords, and to use these grouped keywords as tags with which users can use to filter articles by topic. Finally, our visualizations provide an interactive recommendation system that actively promotes opposing viewpoints, in additional to similar articles with the same viewpoint. Moreover, clustering articles by topic allows our system to identify potential coverage bias, where certain topics may be disproportionately covered from a certain viewpoint.

5 Methods

5.1 Bias Classification

We trained multiple classification models using the Hyperpartisan News Detection Dataset available on HuggingFace [12]. The dataset contains over 750,000 news articles with more fine-grained labels ranging from 0 to 4, with 0 and 4 representing hyperpartisan articles with extreme left or right-wing bias respectively. These labels allow for more nuanced bias classification, identifying not only the political leaning of left or right with labels 1-3, but also whether or not an article is hyperpartisan and should be avoided altogether.

Pre-trained language models including BERT [5] have been shown to provide rich embeddings, and we used the more lightweight DistilBERT [18] to first obtain 768-dimensional feature vectors from the article text. These features were then used to train various classification models including a fully-connected network, as well as naive bayes, logistic regression, passive aggressive classifier, and perception using implementations from the Sci-kit Learn library.

Initially, due to computation limitations, the pre-trained language models were frozen, and only the final classifier was trained using the feature vectors. With this setup, the best performing fully-connected network classifier achieved a training accuracy of 0.8454, but unfortunately only achieved a validation accuracy 0.2891.

Due to the poor performance of these initial methods, we decided to additionally fine-tune a DistilBERT model for sequence classification on the dataset. We use a subset of the train and validation set during fine-tuning to reduce computation requirements. Ideally, unfreezing the model weights and fine-tuning DistilBERT is able to generate richer encodings of the text that would be more easily labeled by a fully-connected classification head.

5.2 Keyword Extraction

We use the PositionRank algorithm to summarize each news article into 5 representative keywords that we can later use in our clustering algorithms. First, stopwords such as "a", "the", "is", etc. are removed from the text, as they tend to provide very little information. The algorithm then uses both the contextual relevance and the frequency of terms within each text to output the keywords. A graph is then constructed from the preprocessed text of the articles, where nodes represent unique word tokens, and edges denote the co-occurrence of pairs of tokens within a defined window of adjacent words. The strength of each connection is weighted by the frequency of co-occurrence, capturing the contextual relationships between words. The scores generated by this algorithm illustrate the significance of each token, with higher scores pointing to greater importance. We then take the the top 5

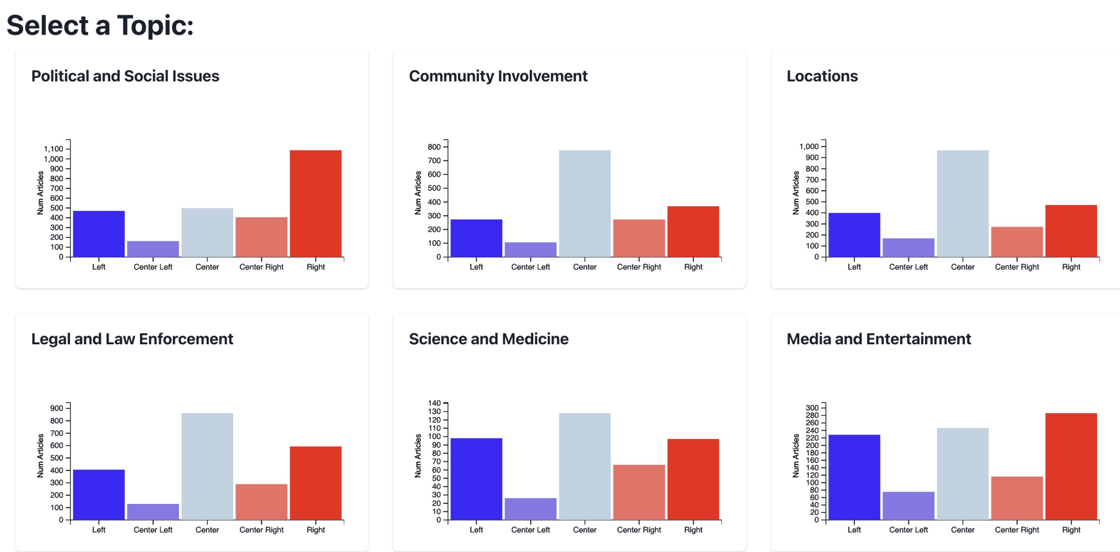

Figure 1: Topic selection interface of NewsMatch.

keywords, which provide a succinct representation that encapsulates the core topics of the article.

5.3 Keyword Clustering

Using the keywords extracted from each article, we additionally group these keywords into overall topic clusters, which can then be used to provide topic tags for each of the articles. We first obtain word embeddings of the keywords using word2vec trained on the Google News dataset, which convert words into 300-dimensional feature vectors while capturing the semantics and relationships between words. We then use clustering methods such as KMeans and DBScan [7] to cluster these feature vectors an obtain topic clusters, where similar keywords are grouped together. Here, the benefit of DBScan is it ability to also identify outlier keywords due to its density-based clustering, so that potentially uninformative keywords can be ignored. The articles are then assigned topic clusters based on their original extracted keywords and which clusters those keywords were assigned. Crucially, this allows our system to assign an article to multiple topic clusters, enabling it to group and filter articles by their topic, creating the backbone of the recommendation system.

5.4 Recommendation System

Our interactive recommendation system, NewsMatch, includes a series of visualizations that conveys articles of similar and opposing viewpoints in a clear and easy to understand manner. Our system begins with a user selecting a keyword for a topic space they are interested in. As they make this selection, they are presented with a histogram for each topic shown in Figure 1, which displays the spread of bias across that topic. Users are able to make a direct comparison between news topics and can also easily identify which topics are either underrepresented or disproportionately covered from different viewpoints. When the user selects a topic, a list of articles is from that topic cluster is displayed.

When the user selects a specific article, they are presented with our recommendation system, where 3 columns will be displayed, presenting articles of the same topic with different bias labels, shown in Figure 2. The leftmost column is a list of articles with a bias rating most similar to the selected article, displayed using simple cards. The middle column is the selected article and specific information about it such as bias, keywords, news source, and more. Finally, the rightmost column will be a list of articles with an opposing bias rating again in the form of cards. This allows users to easily view similar and different viewpoints based on their selected article and expand their depth of knowledge on the topic. Users can additionally select other articles to view additional recommendations based on the chosen article.

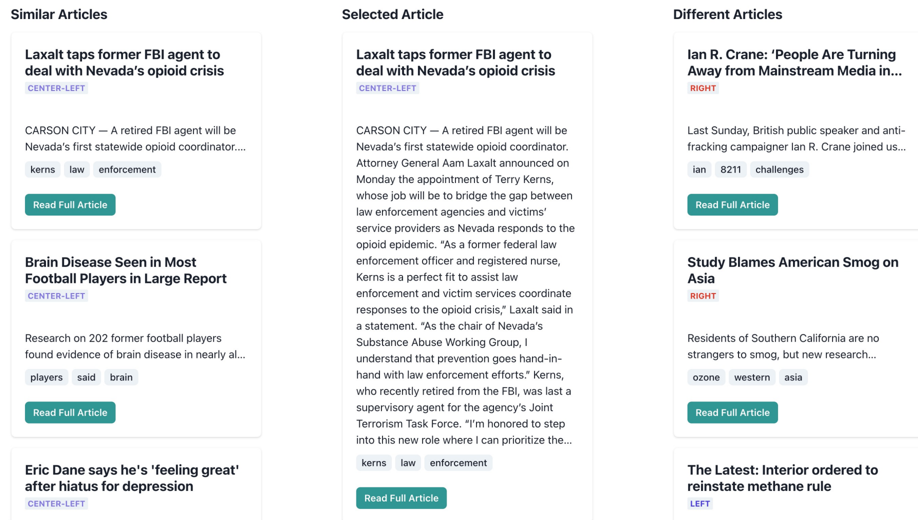

Figure 2: Article recommendation interface of NewsMatch.

6 Experiments and Evaluation

6.1 Bias Classification Models

As previously mentioned, due to computation limitations, our initial experiments use a frozen DistilBERT to obtain embedding vectors from the article text. We trained various classification models using those feature vectors, of which, SGDClassifier (Logistic Regression), Perceptron, MultinomialNB, BernoulliNB, and PassiveAgressive are implemented using the Sci-kit Learn library, and a fully-connected network (FCN Head) is implemented using Huggingface and PyTorch. Here, we record both label accuracy as well as mean absolute error (MAE) as the bias labels are ordered along a scale of 0-4. While our task is a classification problem, MAE still serves to show how far off any misclassifications are. Notably, the traditional classification models struggle to achieve good results, and the best performing model is the fully-connected network for sequence classification, achieving a final validation accuracy of 0.692 and MAE of 0.733 after additional fine-tuning, shown in Table 1.

Despite the improved performance of the FCN Head in comparison to our initial investigation, it still falls short of having good reliability for use in our system. The reason behind the poor performance of these classification models is likely twofold. First, the problem of predicting bias from text is a complex task, requiring an extremely high-dimensional and complex feature vector, which may cause the more simplistic classification models from Sci-kit Learn to struggle. Second, while DistilBERT is pretrained using a corpus from Wikipedia and various books, our corpus of online news articles is slightly different, and likely requires the language model to be fined tuned on our specific corpus in order to capture more nuanced information and patterns needed to properly classify bias. The feature vectors obtained from the frozen language model may not capture the information needed to perform well on the downstream classification task.

We therefore additionally fine-tuned DistilBERT alongside the FCN Head for sequence classification. To work around computation limitations, a small subset of 6400 articles was randomly sampled from the training set at the start of each epoch and used to fine-tune the model. Meanwhile, at the start of training, a subset of 3200 articles was randomly sampled from the validation set, and was used to generate performance metrics for the model. Using a smaller subset of data allowed us to perform some fine-tuning while avoiding large compute times. The results are also shown in Table 1, and the fully fine-tuned model achieves an accuracy of 0.891 and MAE of 0.294, indicating that it is able to correctly identify the bias label most of the time. Even when it is incorrect, its predicted label is usually close or adjacent to the correct label as well.

6.2 Keyword Topic Clustering

The extracted keywords were initially qualitatively evaluated and looked to correctly encapsulate most of the key topics and ideas covered by an article. We then moved forward with both DBScan and

	Model	Accuracy	MAE	Forward Pass Runtime (s)
Frozen	SGDClassifier	0.411	1.457	—
	Perceptron	0.404	1.549	—
	MultinomialNB	0.375	1.360	—
	BernoulliNB	0.245	1.287	—
	PassiveAggressive	0.403	1.471	—
	FCN Head	**0.692**	**0.733**	0.0176
Fine-Tuned	FCN Head	**0.891**	**0.294**	0.0191

Table 1: Validation results for various bias classification models.

KMeans to cluster the keyword embedding vectors obtained using word2vec into large topic tags.

Initial investigations into DBScan with a hyperparameter search show that DBScan is unable to create evenly-sized clusters. Likely due to its density-based nature and high dimensionality of the extracted feature vectors, it either creates one large cluster with the majority of the keywords and many smaller clusters of 1-3 words, or classifies most of the keywords as outliers.

On the other hand, KMeans is able to create relatively evenly-sized clusters. The optimal number of clusters was found to be 14 using the Elbow method, shown in Figure 3. Each of the 14 clusters were then manually assigned overall topics with some experimental use of ChatGPT as follows:

- Political and Social Issues
- Community Involvement
- Names, Organizations, Misc. Terms (Noise)
- Locations
- Misc. Adjectives and Phrases (Noise)
- Legal and Law Enforcement
- Science and Medicine
- Nonsensical Phrases (Noise)
- Media and Entertainment
- Misc. Verbs and Phrases (Noise)
- Nature and Wildlife
- Food
- Finance and Economics
- Objects and Accessories (Noise)

Of the 14 clusters, 5 of them were either non-cohesive, such as "Misc. Adjectives and Phrases", or were not useful for topic assignment, such as "Names, Organization, Misc. Terms" which contained a lot of miscellaneous proper nouns and names. Thees clusters, marked as noise in the list above, were simply discarded and not used in our system. The remaining 9 clusters were then assigned to the articles for use in our recommendation system.

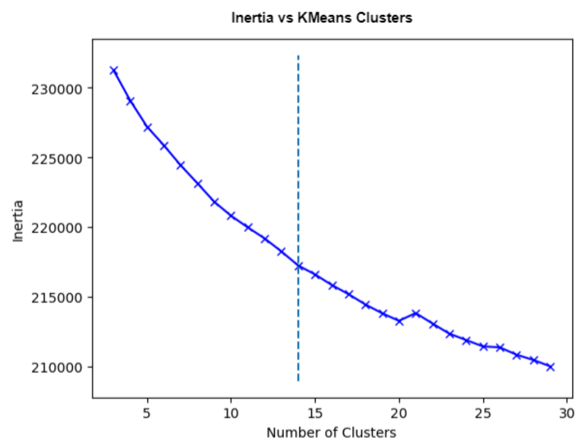

Figure 3: The optimal number of 14 clusters found using the Elbow method.

The clustering provided by KMeans was also evaluated using clustering metrics like Silhouette Score and DB-Index. With 14 clusters, the silhouette score is -0.06973 while the DB-Index is 5.2527, which are relatively poor metrics as higher silhouette score and low DB-index is better. However, this is likely again due to the high dimensionality of the data, causing distance measurements to become less effective. However, despite this, the keyword clusters seem to function well on a qualitative level, and are able to provide broad topic clusters for our recommendation system.

6.3 System Runtime

We performed an analysis of the runtime of our system, especially considering the fact computation must be conducted in real-time if a user inputs their own article text. The runtime of a singular forward pass of the bias classification model is given in Table 1. Where the best-performing fine-tuned model only requires 0.0191 seconds. The relatively fast runtime suggests that our system will easily be able to handle potential custom user input.

We also analyze the time required for our system to group articles by topic and provide recommendations to evaluate the scalability of our system. When processing less than 100k articles at a time, the latency of retrieving counts of bias for each cluster is usually negligible, around 0.2 seconds or less. When the full dataset is used, SQL queries are delayed by less than 10 seconds, and this latency can be reduced by limiting the amount of articles retrieved for each histogram. While there is potential scalability, we'd like to implement further methods in the future to reduce latency when using the full dataset of 500k articles such as caching or pre-computing histogram counts and adjusting as more articles are added to the corpus.

6.4 User Studies

We additionally performed small-scale user studies by asking acquaintances to use our system and having them fill out a four-question survey asking participants to rate their agreement with statements on a scale from 1-5. The survey was sent to 8 participants and the questions are as follows:

- The visualization tool is easy to navigate and use.
- Information such as article bias and topic coverage bias is clearly displayed by the visualization.
- I am more likely to read articles with opposing viewpoints, or articles I would not have considered reading on my own while using this recommendation system.
- How accurate/relevant were the topic clusters and article recommendations?

For questions, at least 60% of users submitted a 4 or 5, indicating that they strongly agreed with the question presented. Importantly, many users felt that our system is effective in promoting new viewpoints, shown in Figure 4. As a result, we believe that our system has been proven to be overall successful in the hands of users. However, there were a few testers responded with a 1 or 2 regarding recommendation accuracy, indicating that our clustering and recommendation algorithm could be improved. In the future, document vector embeddings could be used alongside distance scores to better determine similarity than clustering extracted keywords.

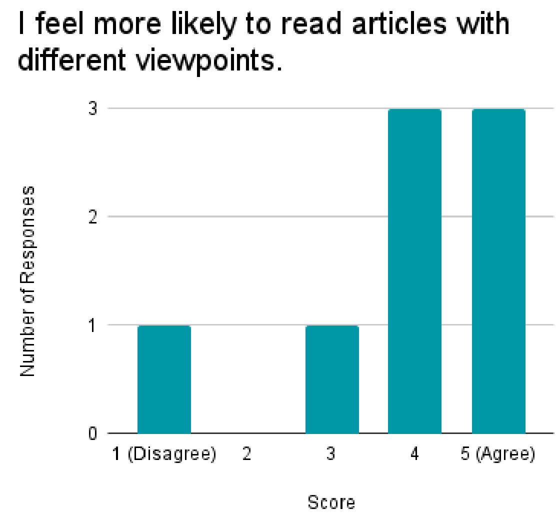

Figure 4: Participant responses to survey question indicate that our system is generally successful in promoting diverse viewpoints.

7 Conclusions and Discussion

We built a fully-fledged tool to identify and inform users of bias across current political news articles. Using a dataset of over 750,000 articles, we trained bias classification models to predict fine-grained bias labels based on article content and additionally perform keyword extraction and clustering to group article by topic. Our system then promotes diverse viewpoints by observing a selected article from the user and providing related articles with similar topics and differing points of view. By providing articles with varying levels of bias, we hope to bring awareness to the polarization of mainstream news, and encourage readers to explore multiple sources to form a well-rounded understanding. Future improvements could consist of implementing retrieval from current news, to keep clusters up-to-date. Overall, our tool provides a lot of value to a user's experience of news browsing, and can be influential in raising awareness of bias in political news.

8 Statement of Cooperation

All team members have contributed a similar amount of effort in this project.

References

[1] Jisun An, Meeyoung Cha, Krishna Gummadi, Jon Crowcroft, and Daniele Quercia. 2012. Visualizing media bias through Twitter. 6, 2 (2012), 2–5.

[2] Brooke E. Auxier and Jessica Vitak. 2019. Factors Motivating Customization and Echo Chamber Creation Within Digital News Environments. *Social Media + Society* 5, 2 (2019).

[3] Ramy Baly, Giovanni Da San Martino, James Glass, and Preslav Nakov. 2020. We Can Detect Your Bias: Predicting the Political Ideology of News Articles. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 4982–4991.

[4] Wei-Fan Chen, Khalid Al Khatib, Henning Wachsmuth, and Benno Stein. 2020. Analyzing Political Bias and Unfairness in News Articles at Different Levels of Granularity. In *Proceedings of the Fourth Workshop on Natural Language Processing and Computational Social Science*. 149–154.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[6] Samantha D'Alonzo and Max Tegmark. 2022. Machine-learning media bias. *Plos one* 17, 8 (2022).

[7] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)*. AAAI Press, 226–231.

[8] Corina Florescu and Cornelia Caragea. 2017. PositionRank: An Unsupervised Approach to Keyphrase Extraction from Scholarly Documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 1105–1115.

[9] Rama Rohit Reddy Gangula, Suma Reddy Duggenpudi, and Radhika Mamidi. 2019. Detecting political bias in news articles using headline attention. In *Proceedings of the 2019 ACL workshop BlackboxNLP: analyzing and interpreting neural networks for NLP*. 77–84.

[10] Nabeel Gillani, Ann Yuan, Martin Saveski, Soroush Vosoughi, and Deb Roy. 2018. Me, My Echo Chamber, and I: Introspection on Social Media Polarization. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. 823–831.

[11] Felix Hamborg, Norman Meuschke, and Bela Gipp. 2020. Bias-aware news analysis using matrix-based news aggregation. *International Journal on Digital Libraries* 21, 2 (2020), 129–147.

[12] Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. 2019. SemEval-2019 Task 4: Hyperpartisan News Detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, Minneapolis, Minnesota, USA, 829–839. https://doi.org/10.18653/v1/S19-2145

[13] Mayank Kulkarni, Debanjan Mahata, Ravneet Arora, and Rajarshi Bhowmik. 2022. Learning Rich Representation of Keyphrases from Text. In *Findings of the Association for Computational Linguistics: NAACL 2022*. 891–906.

[14] Konstantina Lazaridou, Ralf Krestel, and Felix Naumann. 2017. Identifying Media Bias by Analyzing Reported Speech. In *2017 IEEE International Conference on Data Mining (ICDM)*. 943–948.

[15] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order into Text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 404–411.

[16] Archit Rathore, Sunipa Dev, Jeff M Phillips, Vivek Srikumar, Yan Zheng, Chin-Chia Michael Yeh, Junpeng Wang, Wei Zhang, and Bei Wang. 2024. Verb: Visualizing and interpreting bias mitigation techniques geometrically for word representations. *ACM Transactions on Interactive Intelligent Systems* 14, 1 (2024), 1–34.

[17] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text mining: applications and theory* (2010), 1–20.

[18] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv:1910.01108 [cs.CL]

[19] Timo Spinde. 2021. An interdisciplinary approach for the automated detection and visualization of media bias in news articles. In *2021 International Conference on Data Mining Workshops (ICDMW)*. IEEE, 1096–1103.

[20] Timo Spinde, Felix Hamborg, Karsten Donnay, Angelica Becerra, and Bela Gipp. 2020. Enabling news consumers to view and understand biased news coverage: a study on the perception and visualization of media bias. In *Proceedings of the ACM/IEEE joint conference on digital libraries in 2020*. 389–392.

[21] Timo Spinde, Christin Jeggle, Magdalena Haupt, Wolfgang Gaissmaier, and Helge Giese. 2022. How do we raise media bias awareness effectively? Effects of visualizations to communicate bias. *Plos one* 17, 4 (2022).

[22] MG Thushara, Tadi Mownika, and Ritika Mangamuru. 2019. A comparative study on different keyword extraction algorithms. In *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*. IEEE, 969–973.