

# Breaking the Echo Chamber: Analyzing Political Bias in News Media

Yash Rathi  
[yashr@vt.edu](mailto:yashr@vt.edu)

Department of Computer Science  
 Virginia Tech

Seshadri Kakani  
[sckakani@vt.edu](mailto:sckakani@vt.edu)

Department of Computer Science  
 Virginia Tech

Gautham Gali  
[ggali14@vt.edu](mailto:ggali14@vt.edu)

Department of Computer Science  
 Virginia Tech

**Abstract**—Recent years marked an unprecedented flood of information. This constant stream of information overwhelms users and traps them within echo chambers, reinforcing existing opinions while painting dissenting viewpoints as misguided or delusional. The ability to discern politically biased reporting is more critical than ever for fostering informed societies. To this end, we propose a novel tool to classify political bias in news and provide reasoning for the bias to help users gain a balanced perspective of news media. We leverage generalization and emergent abilities of a large language models (LLMs) to generate bias explanations from a labeled political bias dataset – BIGNEWSALGN clustered on news reporting of same story on different biases. We make use of label information and this label explanation to fine-tune for a smaller task-specific LLM. Our tool also integrates various information retrieval techniques to fetch news content to give users plethora of ways to interact with our system. Despite using a very small subset of data, our fine-tune models give solid benchmarks for political bias classification tasks based on generated label. We further discuss how this system empowers individuals to engage with the news they consume, the need for it, and the requirements to build it.

**Index Terms**—Fake news, LLM, Information retrieval

## I. INTRODUCTION

THE news is a crucial public service serving a multifaceted societal role. The news provides people with information that helps with awareness, accountability, public discourse, education, communication, and economic analysis. The news accounts are a timely source of information, delivering updates on local, national and international events, thus keeping individuals informed about their surroundings. It also aids in maintaining awareness of various domains like politics, economics, environment, and technology, among others. The awareness provided by the news allows people to make informed decisions in their personal and professional lives. News organizations and journalists also play a crucial role in upholding accountability by highlighting corruption, injustice, and misconduct and promoting societal transparency. The news also serves an educational purpose, allowing people to understand different cultures, societies, and perspectives.

Which helps foster empathy, understanding, and tolerance. In times of crisis, news outlets provide critical information such as safety instructions and relief efforts—the reasons provided above show why the news is a crucial public service. However, with the excess information available today, there is a significant problem with the bias of many news sources. Among many similar definitions for bias, Merriam-Webster Dictionary defines it as "to give a settled and often prejudiced outlook." [1] or "in a prejudiced manner or with slanted viewpoint", by Lee et.al [2]. Journalism and news bias can distort the public perception of reality, leading to misinformation and polarization. When news organizations or journalists present information from a biased point of view, the facts are not always accurate. If the news is biased, the readers are prone to adopt biased views amplified by social media [3]. The amplification of bias leads to people firmly believing their viewpoint while considering the other viewpoint false and delusional.

People are subject to a daily influx of information from social media, email, and news organizations. People cannot quickly determine the news that is non-biased and which ones are biased. Without understanding which articles are biased, people are misinformed and spread misinformation. Some measures to combat misinformation include community notes and independent organizations that determine bias in news organizations. Community notes on social media, such as X/Twitter, are shown when there is misinformation in the post. The notes explain what the misinformation is and why it is misinformed. Also, they sometimes provide the proper facts.

Additionally, independent organizations can help keep news organizations accountable. Although these countermeasures are helpful, they are not enough. When community notes get added to a post, hundreds or thousands of people have already seen the post. The same problem exists with independent organizations; they are not real-time, and many people do not utilize these resources.

Bias can be of various types. They include:

- Linguistic Bias – bias induced by lexical features.
- Text-level context Bias – words and statements shape the content of article and sway the reader’s perspective.
- Reporting level Bias – bias arising through editorial decisions
- Cognitive Bias – when readers induce bias by their decision to read specific articles and sources to trust.
- Hate Speech – language that degrade, humiliate or

offend targeted groups of people.

- f. Racial Bias – occurs when specific racial groups are portrayed more positively or negatively.
- g. Gender Bias – discrimination or negative portrayal of one gender over other.

While substantial research exists in identifying political bias, to our best knowledge, there’s no tool that exists that also explains the predicted political bias. Lack of reasoning behind bias: a) impacts explainability of bias predicting framework b) makes it harder for users to trust the predicted bias. A model to predict bias and get explanation for the label gives us more perspective into reasoning behind the predicted bias. We also release our dataset with label and explanation for bias label along with models to allow for development for more robust system. LLM are very good NLP tasks because generalization [4] and emergent abilities [5]. But as reported by Bang et. al instruct tuned models also tend to have bias [6] which gives us further motivation to fine tune a task-specific LLM.

In this paper, we propose a news bias system that allows news organizations and readers to explain bias in news. The system explains the bias so it can be accounted for when processing the news.

## II. RELATED WORK

In this section, based on the research on bias detection, fake news identification, and the use of machine learning (ML) for sentiment analysis, we categorize the related work into three sections: Bias Detection and Identification, ML Techniques for Grouping and Sentiment Prediction, and Impact of Fake News on Society.

### A. Political Bias Detection and Identification

Considerable effort has been dedicated to the development of methodologies for detecting political bias in news articles. Baly et. al created a robust dataset of 34,737 articles annotated with political biases and propose an adversarial media adaptation with special triplet loss to achieve SoTA [20]. Gangula et. al create their own dataset and predict news bias based on an attention mechanism applied to headline of a news article [21]. Lu et. al introduce “BiasWatch: A Lightweight System for Discovering and Tracking Topic-Sensitive Opinion Bias in Social Media” which can detect bias in wide range of news sources [22]. Fan et.al introduce a methodology to analyze bias in news articles at different granularity level and further studies the bias distribution [23]. Raza et. al introduces Dbias, an open-source Python package which can take any text to determine if it is biased. It also detects biased words in the text, masks them, and suggests a set of sentences with new words that are bias-free or at least less biased [24]. Liu et. al further propose “POLITICS” which sets a new state of art for ideology prediction and stance detection tasks [25]. None of the bias detection tools and approaches to our best knowledge give explanation behind predicted bias and only focus on: 1) debiasing the bias news article 2) predicting the bias of news article 3) stance and ideology detection.

### B. Text Classification using LLM

Large Language Models (LLMs) such as GPT-3 have shown remarkable success in various natural language processing tasks [7]. However, their performance in text classification tasks significantly underperforms compared to fine-tuned models. This underperformance is attributed to two main reasons:

**Lack of Reasoning Ability:** LLMs struggle with complex linguistic phenomena such as intensification, contrast, and irony.

**Limited Tokens in In-Context Learning:** The number of tokens that can be used in in-context learning is limited.

To address these challenges, Clue And Reasoning Prompting (CARP) was introduced by Sun et al. CARP adopts a progressive reasoning strategy tailored to address the complex linguistic phenomena involved in text classification [8]. It first prompts LLMs to find superficial clues (e.g., keywords, tones, semantic relations, references, etc), based on which a diagnostic reasoning process is induced for final decisions. To address the limited-token issue, CARP uses a fine-tuned model on the supervised dataset for kNN demonstration search in the in-context learning. CARP has shown impressive abilities on low-resource and domain-adaptation setups.

Another approach is TELEClass [9], which enriches the label taxonomy with class-indicative topical terms mined from the corpus to facilitate classifier training. It also utilizes LLMs for both data annotation and creation tailored for the hierarchical label space.

In the legal domain, an empirical study compared a standard, pretrained DistilBERT model and a fine-tuned DistilBERT model for text classification [10]. The study found that fine-tuning the model using domain-specific data from real-world legal matters improves the performance of LLM text classifiers.

### C. Large Language Models

Large Language Models (LLMs) have been a significant area of research in the field of Natural Language Processing (NLP). They are built on transformer [11] architecture, allowing them to capture complex language patterns and relationships between words or phrases in large-scale text datasets. LLMs can be categorized into three main types: encoder-decoder, causal decoder, and prefix decoder [12][13]. Each type has its advantages and has been used in different LLMs but almost all current SOTA models are based on GPT like decoder only architecture. Llama and Gemini are two state-of-the-art LLMs that we specifically used for finetuning. Llama [14], based on decoder only architecture, introduced by Meta, is a state-of-the-art openly available family of LLM. It demonstrates state-of-the-art performance on a wide range of industry benchmarks. Gemma is a family of lightweight, state-of-the-art open models from Google. It utilizes a dual encoder, employing separate encoders to process the context and response in a conversation<sup>3</sup>. Gemma models achieve exceptional benchmark results at its 2B and 7B sizes, even outperforming some larger open models [15]. Qwen1.5 are series of LLM from Alibaba with huge context length of 32k, SOTA performance on various benchmarks and multilingual capabilities [16]. Qwen1.5 72b was used to generate bias explanations.

#### D. Finetuning Large Language Models

Fine-tuning Large Language Models (LLMs) involves adjusting the parameters of a pre-trained model to adapt it to a specific task. Zhang et al. conducted systematic experiments to understand the inductive biases of different fine-tuning methods. They considered two types of fine-tuning – full-model tuning (FMT) and parameter efficient tuning (PEFT, including prompt tuning and LoRA), and explored their scaling behaviors in the data-limited regime where the LLM model size substantially outweighs the fine-tuning data size. Their findings suggest that the optimal fine-tuning method is highly task- and fine-tuning data-dependent [17].

One of the methods explored by Zhang et al. is Low-Rank Adaptation (LoRA), which freezes the pre-trained model weights and injects trainable rank decomposition matrices into each layer of the Transformer architecture, greatly reducing the number of trainable parameters for downstream tasks. This approach has been shown to perform on-par or better than fine-tuning in model quality on RoBERTa, DeBERTa, GPT-2, and GPT-3, despite having fewer trainable parameters, a higher training throughput, and, unlike adapters, no additional inference latency [18].

Building upon the concept of LoRA, Dettmers et al. introduced QLoRA, an efficient fine-tuning approach that reduces memory usage enough to fine-tune a 65B parameter model on a single 48GB GPU while preserving full 16-bit fine-tuning task performance<sup>1</sup>. QLoRA backpropagates gradients through a frozen, 4-bit quantized pre-trained language model into Low Rank Adapters (LoRA). QLoRA introduces a number of innovations to save memory without sacrificing performance: 4-bit NormalFloat (NF4), a new data type that is information theoretically optimal for normally distributed weights; double quantization to reduce the average memory footprint by quantizing the quantization constants; and paged optimizers to manage memory spikes [19].

### III. PROPOSED APPROACH

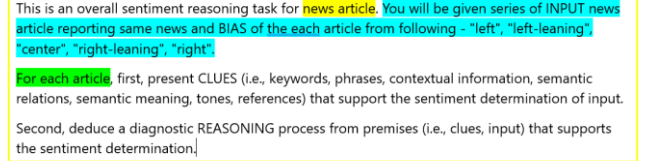
#### A. Dataset Selection

The BIGNEWS dataset is a large-scale dataset introduced by Liu et al. in "POLITICS". Here are some key details about the dataset: The dataset contains 3,689,229 English news articles on politics gathered from 11 United States (US) media outlets and cover a broad ideological spectrum. We specifically choose the BIGNEWSBLN variant of BIGNEWS since it is clustered dataset with around ~1 million stories. Each cluster has reporting of story from each ideological side i.e. left, center, right. Furthermore, we preprocess these 1 million stories to around 35k stories in final dataset. We achieve that by: 1) First we relabel dataset to get labels "Lean Left" and "Lean Right". This is possible since authors of POLITICS provide a spectrum for ideological leaning of organizations in their original work. 2) Ensure each story cluster has article from all 5 biases including the two new added labels. This brings down total eligible clusters to around ~180k stories. 3) We then apply common preprocessing techniques like outlier removal, keeping average word count of all articles to around 4000 words

or around 20000 words per cluster all biases included to ensure that articles fit the context length of LLM used for bias explanation generation. Our final preprocessed dataset consisted of around ~35k clustered stories or 175k news articles. However, due to resource limitations we only pick 500 clusters or 2000 news articles for train set to train our model. We found it was cost prohibitive to generate explanations for so many news articles.

#### B. Generating Bias Explanation

For generating bias explanation text, we use best openly available LLM for the task. Generating data from larger model to fine-tune smaller model is already a well-established technique. At time of working on this, we found the model to be Qwen1.5 Chat with 70 billion parameter count. In this context, we mean "best model" as a model with highest rating on LMSYS Leaderboard [26]. We found LMSYS to be reliable because it is hard to manipulate the benchmarks and authors thoroughly cover their methodology in published work. For prompting, we use a modification of CARP prompt tailored for generating Clues and Reasoning for news bias detection.



This is an overall sentiment reasoning task for news article. You will be given series of INPUT news article reporting same news and BIAS of the each article from following - "left", "left-leaning", "center", "right-leaning", "right".

For each article, first, present CLUES (i.e., keywords, phrases, contextual information, semantic relations, semantic meaning, tones, references) that support the sentiment determination of input.

Second, deduce a diagnostic REASONING process from premises (i.e., clues, input) that supports the sentiment determination.

Fig 1: Modification of CARP Prompt

We provide LLM with reporting of news article from all 5 biases in a single prompt. This is possible because model we use for reasoning i.e. Qwen1.5 72B has large context window of 32k tokens. This also helps LLM generate more accurate Clues and Reasoning for given article. We also do manual and automated checks on generated explanations to make sure clues and reasoning are present for each article.

#### C. Formatting for Fine-tuning

We use the Alpaca format to fine-tune our candidate LLMs: Llama 3 7B and Gemma 7B. Alpaca is mainly used for instruction tuning for task-specific outputs. It was introduced by Stanford's Alpaca project [27]. We use Alpaca to adapt the model to follow instructions by learning from the instruction-answer pairs in the dataset. Thus, we format our preprocessed dataset to pair of instruction answer pairs.

#### D. Inferencing

While inferencing, we similarly use Alpaca format. Response is masked from model. Fine-tuned model completes response and includes bias analysis.

#### E. QLoRA Training and Quantization

We train a QLoRA on Llama3 7b and Gemma 7b, respectively. We used parameters suggested by Dettmers et. al. We train for 1 epoch with rank 16 and alpha 16 for target modules ["q\_proj", "k\_proj", "v\_proj", "o\_proj", "gate\_proj",

"up\_proj", "down\_proj". After training QLoRA adapter, we merge it with base model to get the final fine-tuned model. All models finetuned by us are openly available on HuggingFace. All merged finetuned models were originally saved in FP16 (half precision). But since it's not feasible to run FP16 variants, we further quantize the models to Q5\_K\_M and Q8\_0. Quantization causes minimal loss in perplexity compared to FP16 but considerable memory savings and much faster inference speeds [28].

Fig 2: Sample dataset formatted in Alpaca format for training.

Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.

#### ### Instruction

This is an overall sentiment reasoning task for a news article. You will be given a INPUT news article. First, find the **BIAS** of the news article (i.e., "left", "lean left", "center", "lean right", "right"). Second, present **CLUES** (i.e., keywords, phrases, contextual information, semantic relations, semantic meaning, tones, references) that support the sentiment determination of input. Finally, deduce a diagnostic **REASONING** process from premises (i.e., bias, clues, input) that supports the sentiment determination.

#### ### Input

{news article}

#### ### Response

{bias explanation}

## IV. SYSTEM ARCHITECTURE

In this section, we outline the design of proposed system.

### A. User Interface and Website

Our system features a sophisticated, web-based user interface designed to ensure user-friendly navigation and efficient interaction, underpinned by a robust backend architecture. The backend is powered by the Python Flask API, with MongoDB serving as the database for storing and retrieving large volumes of text data efficiently. Beautiful Soup is employed for web scraping tasks, enabling the extraction of article content directly from URLs provided by users. The user interface is meticulously organized into three main sections, each designed to cater to different user needs: query search, article text search, and URL search. In the query search section, users can search our MongoDB database using specific queries. Here, the system uses advanced information retrieval techniques such as TF-IDF vectorization to find and display articles that are most relevant to the entered query. When queries result in few or no results, the system extends the search to external sources via Google, ensuring a thorough and comprehensive search experience. For URL searches, users can submit URLs from specific, supported news sources like CNN, Fox News, and three other major outlets. The system utilizes Beautiful Soup to scrape the article content

from these URLs. Once extracted, the content is analyzed by a sophisticated Large Language Model (LLM) to determine the political bias of the article. The model not only classifies the bias as left, right, or center but also provides detailed reasoning behind its classification, enhancing transparency and user trust. Similarly, the article text search allows users to directly input the text of a news article. This text is processed and analyzed by the same LLM, which assesses and returns

### BIAS - LEFT

#### CLUES:

1. "knee-jerk opposition and obstruction just demonstrates that Republicans haven't been chastened by the 2012 election"

2. "making a ...

#### REASONING:

The article presents a clear critique of the Republican opposition to Obama's nominees...

Fig 3: Bias and Bias Analysis from fine-tuned model

the political bias along with comprehensive clues and reasoning that explain the basis of its analysis. The front end of the system is developed using HTML and CSS, ensuring that the interface is not only functional but also visually appealing and responsive to various devices and screen sizes. The design facilitates easy navigation and an engaging user experience. Through each step of the interaction—from inputting data, through processing and analysis, to the presentation of results—the system is designed to provide rapid, accurate bias classifications and detailed explanations.

Article Text	Result Obtained from LLM
<p>4 law enforcement officers were killed in shooting at a home in Charlotte, North Carolina. 4 other officers are hospitalized</p> <p>Four officers were killed in a shooting while attempting to serve a warrant at a home in Charlotte, North Carolina, including one deputy US marshal and two local task force officers, authorities say.</p> <p>Four other law enforcement officers were shot during the incident, Charlotte-Mecklenburg Police Chief Johnny Jennings said Monday evening.</p> <p>An internal law enforcement memo reviewed by CNN shows officers are looking for the person who bought the firearm used in the shooting to potentially bring federal charges.</p> <p>Jennings said during a Tuesday news conference the investigation is active and ongoing.</p> <p>Evidence from the residence is still being processed, he said, adding the amount of evidence that will be collected is expected to be "well over 100</p>	<p>BIAS - LEAN LEFT</p> <p>CLUES:</p> <ol style="list-style-type: none"> <li>1. Mention of the suspect's criminal history, including possession of a firearm by a convicted felon.</li> <li>2. The internal law enforcement memo seeking charges against the gun buyer.</li> <li>3. The focus on the investigation and the evidence collected.</li> <li>4. The attention to the tragedy and the impact on the law enforcement community.</li> </ol> <p>REASONING:</p> <p>The article highlights the shooter's criminal past and the investigation around the gun used, which suggests a more analytical approach. The emphasis on the officer's heroism and the tragic nature of the event, while not as emotionally charged as the left bias, still portrays the incident as a critical issue that requires attention from authorities.</p>

Fig 4: UI output after the bias detection

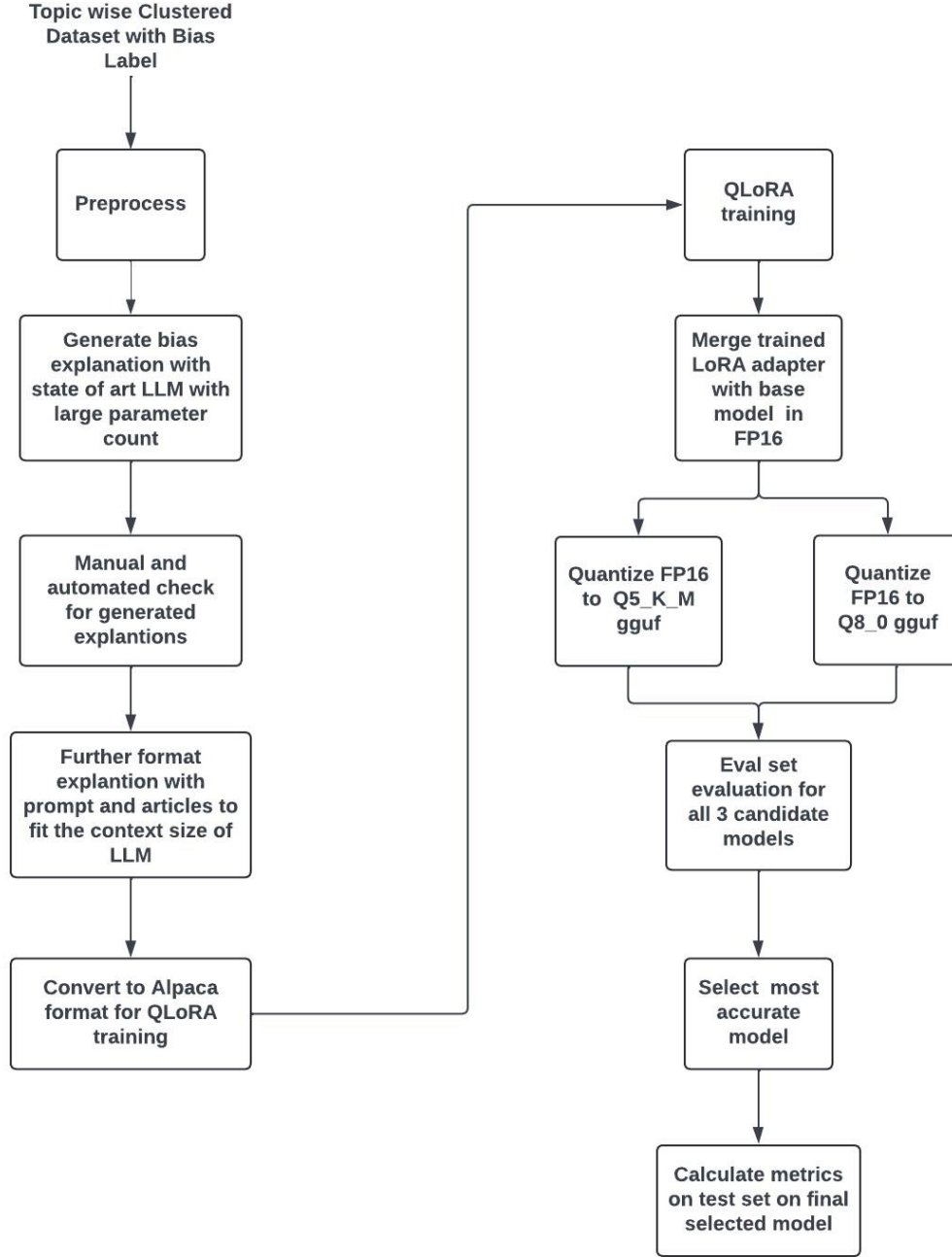


Fig 5: Proposed Approach

### B. Bias Detection and Analysis Module

This module consists of LLM we fine tuned for bias detection and explanation. Merged model is served through a model server optimized for long queries and latency. This module takes input news article from any of 3 available interaction choices for user: 1) their own article 2) webpage link or 3) query based retrieval and returns bias of news article along with clues and reasoning for it.

### C. Information Retrieval (IR) Module

The information retrieval system empowers users to efficiently search our extensive database. Users provide a query to the system, which then utilizes classical information retrieval

methods to retrieve the most relevant documents. Subsequent section will delve into the information retrieval system, focusing on how we build our database and the methods we employ for retrieval.

### D. Information Retrieval (IR) Datasets Used

The information retrieval system relies on two datasets to populate our database: the "All the News" dataset and Common Crawl. The "All the News" dataset is a substantial collection of news articles from various reputable sources. Comprising approximately 200,000 articles, this dataset includes content from prominent publications like The New York Times, The Washington Post, CNN, and Fox, covering a diverse array of



topics. The dataset structure features metadata for each article, including the title, publication date, author, article text, publication, and URL. However, this dataset only contains data up to 2020, making it somewhat outdated. It's worth noting that a newer version of "All the News" exists, which contains approximately 2.6 million articles. Although this dataset was not used in this project, it could be integrated later. To address the limitation of outdated information in "All the News" and include up-to-date information, we incorporated data from Common Crawl. Common Crawl is an open repository of web crawl data that contains petabytes of data gathered from the web, including web pages, metadata, and text content from across the internet. The repository is organized into monthly snapshots containing compressed files with raw web data in WARC format. These snapshots, stored in an AWS S3 bucket, are updated monthly. Common Crawl remains relevant due to its continuous updates, which occur a couple of times each month. By fetching the latest WARC file from the S3 bucket, we extract and process fresh content for our database. However, working with Common Crawl presents challenges, particularly in handling the WARC file format. To facilitate this process, Common Crawl provides scripts to download the latest file, uncompress it, and extract the article data and metadata.

#### E. IR System Database Creation and Preprocessing

The database creation process involves several crucial steps. We begin by sourcing articles from the aforementioned datasets. The first step involves reading the new articles from the CSV file. Key columns include the title, author, publication, date, URL, and content. Before storing this information in the database, we preprocess the title and content to remove stop words, punctuation, and publisher names, and we use the Natural Language Toolkit (NLTK) to lemmatize the words. This preprocessing reduces the size of the title and content, enhancing the efficiency of subsequent processes.

Lemmatization is crucial during preprocessing as it allows the text to become smaller while retaining the content. It's preferable to stemming because it preserves the context of words. For instance, stemming the word 'caring' would return 'car', while lemmatizing the word 'caring' would return 'care'. Preprocessing also improves the final results by focusing only on relevant words and making the content smaller, which accelerates the search function.

Next, we utilize scikit-learn's 'TfidfVectorizer' to vectorize the title and content. These vectors are critical for the search process, so we store them in the file system. The search function later uses this stored vectorizer to vectorize user queries. We experimented with other vectorizers but chose 'TfidfVectorizer' because it had a vector limit of 500, which sped up the search process and provided uniform vector sizes for most articles. We also perform sentiment analysis on the article text using NLTK's 'SentimentIntensityAnalyzer', yielding scores for positive, negative, and neutral sentiment. The sentiment analysis was the initial version of bias detection, as articles with neutral scores tend to be more factual and unbiased, while those with high positive or negative scores might exhibit bias. After preprocessing, vectorization, and sentiment analysis, the processed data is stored in a MongoDB

database, with each article accompanied by relevant details, its vector, and sentiment scores.

#### F. Search Functionality

The final aspect of the information retrieval system is the search functionality. The system accepts a user query and processes it for retrieval. The query is first preprocessed to remove stop words and punctuation, while also lemmatizing the words, ensuring a standardized form for comparison. Next, the system loads the 'TfidfVectorizer' trained on the article titles from a serialized file, transforming the preprocessed query into a vector. The system then connects to the MongoDB database to fetch the article vectors, comparing the query vector with the title vectors of all articles using cosine similarity. We used cosine similarity because it was easy to implement, well-known, and provided a way to rank the results based on relevance. The similarity scores are used to rank the articles in order of relevance. We initially experimented with searching based on the entire article content and based on the title plus the first 100 words of the article, but these approaches resulted in slower search times. Searching based solely on the title expedites the process while still yielding relevant results. Although this impacts relevance, it was a trade-off we were willing to make for increased response speed. The top 10 articles, based on their similarity scores, are selected and returned to the user, ensuring a fast and responsive search experience.

The information retrieval system leverages classical information retrieval techniques to create and search a database of articles. Utilizing the "All the News" and Common Crawl datasets, the system encompasses a wide variety of relevant content. The database creation involves preprocessing, vectorization, and sentiment analysis. The search functionality, using cosine similarity, focuses on title-based retrieval to maximize speed and relevance. This approach ensures that users receive timely and relevant responses to their queries.

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

#### G. Web Crawl System

The URL scraper component of the information retrieval system is designed to fetch and extract data from various news websites based on a given URL. The system operates by loading the URL provided by the user, scraping the data and title from the corresponding webpage. This functionality supports multiple news sources, including CNN, Fox News, The New York Times, The Washington Post, and USA Today.

The scraper employs the 'requests' library to fetch the HTML content from the provided URL and then uses BeautifulSoup to parse and extract the relevant information. The system includes specific functions for each supported news website to handle

their unique structures. Each function extracts the headline and body content, cleans the text to remove unwanted characters and normalize whitespace, and returns the cleaned data in a dictionary format.

The scraper also includes a function that detects the news site based on the URL's domain name, directing the scraping to the appropriate function. The resulting data is formatted as a JSON string, including the URL, title, and body of the article. If the

URL doesn't match any of the supported domains, the system returns an error message indicating that the site is not supported.

In addition to standard news sources, the scraper has been adapted to handle unconventional and emerging news platforms. As online news becomes increasingly diverse, our system has evolved to support a broader range of news domains, ensuring comprehensive coverage and up-to-date information for users.

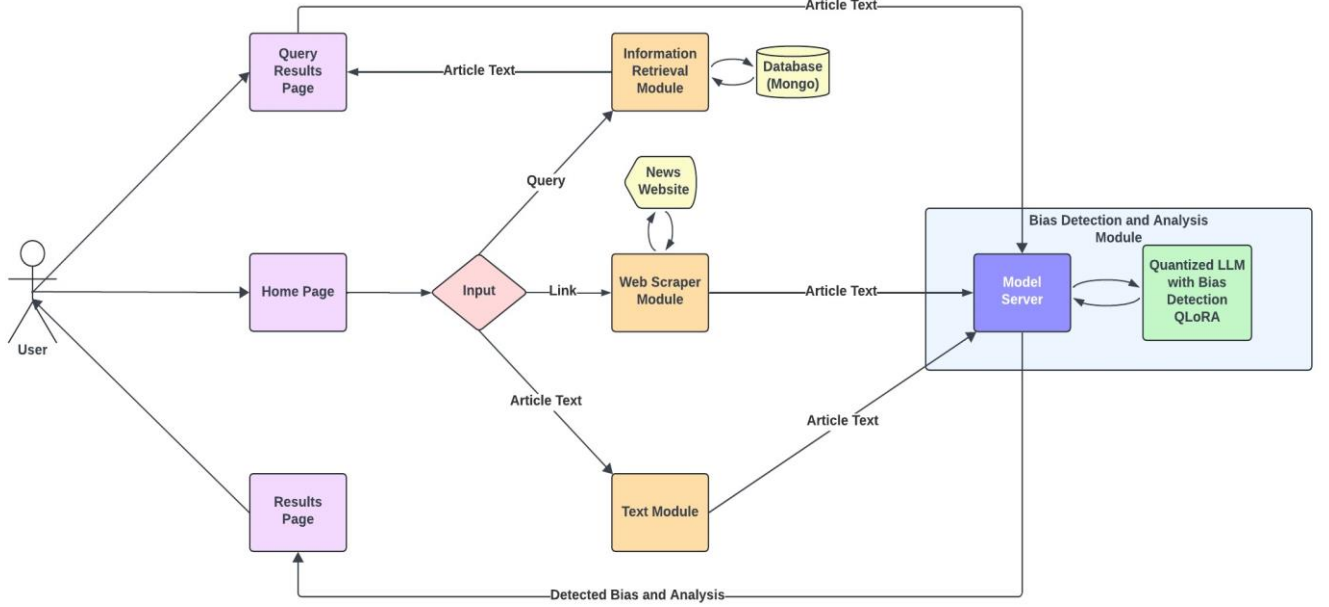


Fig 6: System Diagram

## V. EVALUATIONS

In this section, we present our evaluation methodology and metrics.

### A. Model Candidates

As mentioned in previous section, our approach involves fine-tuning two model i.e. Llama 3 7b and Gemma 7b merged with QLoRA trained for bias detection. After getting original models in FP16, we get two more quantized versions i.e. Q5\_K\_M and Q8\_0. After we do quantization for both aforementioned models, we get total of 6 candidate models for each language architecture. We run these 6 candidate models through our evaluation set.

### B. Candidate Models Evaluation

We run 6 candidate models through our evaluation set consisting of 300 articles or 60 clustered stories. We opt for train-eval-test split as it is a well-established strategy for benchmarking deep learning models. Secondly, having an eval set saves us from running each model through test set. This is important because running LLM can be very computationally expensive and cost prohibitive. Primary purpose of eval set is to pick best candidate model and then further test it on test set to get insight on model performance. We also show evaluation of labels “left”, “center”, “right” as these are most prevalent

labels used in existing research work along with more granular 5 bias labels that we devised.

### C. Selected Model Testing

After selecting best model based on accuracy from eval set, we picked Gemma 7b Q5\_K\_M for use in our system. Gemma 7b Q5\_K\_M consistently outperforms other candidate models in both traditional “left”, “center”, “right” bias identification task and “left”, “lean left”, “center”, “lean right”, “right” bias identification task. Our test set consisted of total 700 articles or 140 story clusters. We first present accuracy of selected model and then F1- micro and F1-macro scores along with F1 scores for individual labels. We specifically chose F1 scores because its established metric used by existing political bias methods and techniques.

Labels – Left, Lean Left, Center, Lean Right, Right

#### Llama3 8b

Base - [unsloth/llama-3-8b-bnb-4bit](#)

	Q5_K_M	Q8_0	FP16
Base + QLoRA	34.50%	34%	37%
Base	25.50%	33.50%	31%

#### Gemma 7b

Base - [unsloth/gemma-7b-bnb-4bit](#)

	Q5_K_M	Q8_0	FP16
Base + QLoRA	54.50%	47.50%	49.50%
Base	27.50%	23.50%	24%

Fig 7: Evaluation Results for Candidate Models 1

Labels – Left, Center, Right

#### Llama3 8b

Base - [unsloth/llama-3-8b-bnb-4bit](#)

	Q5_K_M	Q8_0	FP16
Base + QLoRA	55%	52.50%	61%
Base	49%	51%	53%

#### Gemma 7b

Base - [unsloth/gemma-7b-bnb-4bit](#)

	Q5_K_M	Q8_0	FP16
Base + QLoRA	67.50%	62%	64%
Base	47.50%	35.50%	37.50%

Fig 8: Evaluation Results for Candidate Models 2

#### Gemma 7b Q5\_K\_M (Test Set)

Labels – Left, Lean Left, Center, Lean Right, Right

Accuracy – 51.20%

	Q5_K_M
F1-micro	51.20%
F1-macro	51.77%

Labels – Left, Center, Right

Accuracy – 65%

	Q5_K_M
F1-micro	65.00%
F1-macro	61.10%

Fig 9: F1 scores for Gemma 7b Q5\_K\_M

## VI. ABLATION STUDY

As evident from metrics in Fig 5 and Fig 6, fine-tuning of models with QLoRA has led to improvements in model accuracy across tested metrics, particularly in the Gemma 7b

model. The Gemma 7b model

Gemma 7b Q5_K_M	LEFT	LEAN LEFT	CENTER	LEAN RIGHT	RIGHT
F1 Score (average=None)	62.50%	49%	36%	61%	47%

Gemma 7b Q5_K_M	LEFT	CENTER	RIGHT
F1 Score (average=None)	73%	36%	78%

Fig 10: F1 Scores for each class (Gemma 7b Q5\_K\_M)

showed remarkable improvements, across all quantization: Q5\_K\_M, Q8\_0, and FP16 metrics, suggesting that QLoRA was effective in enhancing the model’s political bias detection capabilities compared to Llama3 8b. The least improvement was observed in the Q8\_0 quantization of the Llama3 8b model. Overall, finetuning LLM proved to be an effective method for detecting political bias, with Gemma 7b based fine-tuned models showing most promise.

#### Llama3 8b Model Accuracy and Improvements

Configuration	Q5_K_M	Q8_0	FP16
Base	25.50%	33.50%	31%
Base + QLoRA	34.50%	34%	37%
Improvement	+9.00%	+0.50%	+6.00%

Fig 11: Llama3 8b Ablation Study

#### Gemma 7b Model Accuracy and Improvements

Configuration	Q5_K_M	Q8_0	FP16
Base	27.50%	23.50%	24%
Base + QLoRA	54.50%	47.50%	49.50%
Improvement	+27.00%	+24.00%	+25.50%

Fig 12: Gemma 7b Ablation Study

## VII. LIMITATIONS

While use of LLMs present a new novel approach to political bias detection and analysis, there are still many limitations that need to be addressed.

First, we don’t have any metrics to evaluate the bias analysis generated by LLM. Its well established that LLMs are prone to hallucination and generating incorrect information. One way to solve this would be to use human in loop evaluation and correct the bias analysis before fine-tuning step of LLM.

Secondly, we only used a very small subset of dataset due to constrains on computational resources. While LLMs have become more accessible than ever, running a large parameter model remains expensive. Despite advancements in finetuning techniques, QLoRA still requires a powerful hardware. LLM landscape is continuously evolving and rapid advancements in field would definitely change this scenario.

Finally, more robust datasets and benchmarks for political bias detection are needed to have a comprehensive



understanding of model performance. One way to solve this would be to use MBIB introduced by Wessel et. al [29]. Still considerable gap remains and better tools and approaches are needed especially in this age of generative AI.

### VIII. FUTURE WORK

For future work, we would like to use more articles for fine-tuning the model. This is to check if model learns more and gets better at detecting political bias.

Another direction we would like to take is to explore detecting and analyzing other types of biases i.e. gender bias or cognitive bias. More often than in real world setting biases are not mutually exclusive and given emergent abilities of LLMs they could get even better at detecting political biases based on additional learning from detecting other kind of biases.

Other direction would be to debias the biased news articles using LLMs. This research would have lot of practical application and its something we would like to work on in future.

Our approach can be generalized to any LLM model and bias type, not only political bias. As better models and fine-tuning techniques get available, bias detection capacities of model will naturally grow. We would like to continue working considering this and are excited to see what we could achieve with better methodologies.

### IX. ADDITIONAL WORK

During our initial work on problem statement, we also explored various machine learning models for news dataset. Ultimately, our main goal was to use LLMs to classify and analyze news bias. But nonetheless, training these models on our preprocessed dataset was a good way for us to gauge and understand the various approaches for political bias detection.

#### A. Text Cleaning

As mentioned in previous section, our approach involves fine-tuning two model i.e. Llama 3 7b and Gemma 7b merged with QLoRA trained for bias detection.

The preprocessing begins with normalizing and simplifying the text data:

**Non-letter and Non-number Removal:** Use regular expressions to remove characters that are neither letters nor numbers, except for a few punctuation marks. This is crucial to eliminate potential noise in the text data that could distract the learning algorithms.

**Lowercasing:** Converting all text to lowercase ensures that the algorithm treats words with different capitalizations (like "The" and "the") as the same word.

**Stopword Removal:** Common words that are unlikely to contribute to the detection of unreliable news (such as "and", "is", "in") are removed using the NLTK library's list of English stopwords. This step helps focus the model's attention on more meaningful words.

The cleanup function further strips any remaining punctuation from the text.

#### B. Constructing Labeled Sentences

We format the cleaned text data into a structure suitable for training a Doc2Vec model. We wrap the text of each article into LabeledSentence objects, which pair the text with a unique label. This setup is necessary for the Doc2Vec algorithm, which requires labeled examples to learn vector representations.

#### C. Vectorization with Doc2Vec

Doc2Vec is employed to convert text into numerical representations that capture semantic meanings of the words in context. We do following tasks:

**Handles Missing Values:** Rows with missing text data are identified and removed to ensure the quality and completeness of the training data.

**Vectorization:** The cleaned text is used to build and train a Doc2Vec model. This model transforms the text into fixed-length vectors that serve as input features for machine learning models.

**Data Splitting:** The dataset is split into training and testing sets, maintaining the distribution of data as it trains and evaluates the model.

#### D. Saving Processed Data

We streamline the cleaning, shuffling, and splitting of the data, preparing it for the model training phase. It ensures that the data is randomly shuffled to prevent any order bias during training. The split datasets are then saved locally, allowing for reproducibility and ease of access during model training.

#### E. Saving Processed Data

The preprocessing steps were implemented in Python, utilizing libraries such as pandas for data manipulation, nltk for natural language processing, and gensim for implementing the Doc2Vec model. This preprocessing is critical for the subsequent machine learning tasks, so that the data fed into the models is of high quality and well-suited for learning patterns related to the reliability of news articles.

#### F. Models

##### a. SVM:

The Support Vector Machine (SVM) powerful classification technique first introduced by Vladimir N. Vapnik and Alexey Ya. Chervonenkis in 1963. Originally designed for linear classification, the original SVM was limited in handling the non-linear datasets that are common in real-world applications. This limitation was significantly overcome in 1992 when Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik introduced the kernel trick, enabling SVM to perform non-linear classification. The kernel trick transforms the original feature space into a higher dimension where a linear separator might be found, thus vastly enhancing the applicability of SVMs to a broad range of classification tasks. In our project, we employ the Radial Basis Function (RBF) kernel for the SVM. The RBF kernel is particularly suited for scenarios where the relationship between class features is not linear. We utilize the Python

package CVXOPT for this. CVXOPT provides tools for solving quadratic programming problems that arise in the implementation of SVM with the RBF kernel, ensuring that we efficiently find the optimal hyperplane for our classification needs.

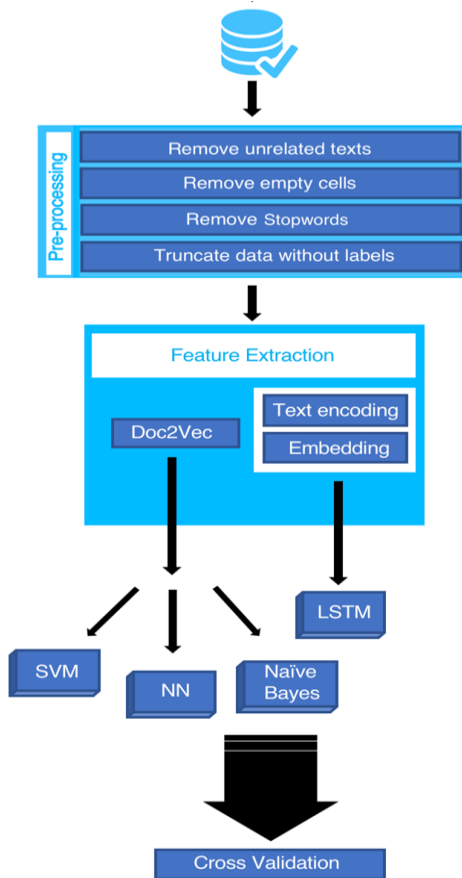


Fig 13 – Additional Work for Political Bias Detection

b. Feed-forward Neural Network: Feed-forward neural networks represent a fundamental architecture in machine learning and have proven particularly effective in numerous natural language processing (NLP) tasks. Contrasting with traditional linear models such as SVMs and logistic regression, which might struggle with the complex and nonlinear nature of language data, neural networks offer a more dynamic and powerful approach. TensorFlow Implementation: This configuration consists of three hidden layers, each containing 300 neurons. TensorFlow provides a flexible and comprehensive ecosystem for defining, training, and validating deep learning models with fine-grained control over model architecture and training parameters. Keras Implementation: Keras, known for its user-friendly and modular approach, is used to set up a similar neural network structure but with different layer sizes: two layers of 256 neurons each followed by a third layer of 80 neurons. Keras simplifies the process of model creation and experimentation, allowing for rapid prototyping. Both implementations use the

Rectified Linear Unit (ReLU) as the activation function for hidden layers. ReLU is chosen for its efficiency and effectiveness in promoting non-linear learning.

c. LSTM: The LSTM network, first introduced by Hochreiter and Schmidhuber in 1997, is a type of recurrent neural network (RNN) specifically designed to handle sequence prediction problems. This architecture is particularly adept at managing long-range dependencies in sequential data, a common challenge in text processing where the order of words is crucial for understanding the overall meaning. Text data, such as news articles, is inherently sequential and rich in context, where not only the presence of words but also their order and proximity can significantly alter meanings. Traditional models like Doc2Vec, which compress an entire document into a single vector, lose this order information. In contrast, LSTMs maintain this sequential integrity by processing text as a series of inputs and effectively capturing temporal dynamics. The LSTM unit's architecture is designed to mitigate the vanishing gradient problem common in traditional RNNs, allowing it to learn from data points that are far apart in the input sequence.

- d. Naïve Bayes: To establish a baseline for the accuracy of our predictive models, we employed the Gaussian Naïve Bayes classifier, a well-known algorithm in the realm of machine learning for its simplicity and effectiveness in probabilistic classification. This classifier is part of the scikit-learn library, which provides robust implementations for a variety of machine learning algorithms. The Gaussian Naïve Bayes classifier operates under the assumption that all features are conditionally independent given the class label. This assumption, while simplistic, allows for the efficient computation of the likelihood of each class based on the input features. The model applies Bayes' Theorem to compute the posterior probability of each class, making a prediction based on the class with the highest posterior probability.

Doing additional work using different ML models gave us a great insight on dataset we were using for finetuning. We also found out some models were overfitting on data, which help us understand how certain portions of data might be causing model to overfit. We took all these insights into account when working with LLM training. These also helped us save considerable resources, since LLM training is expensive and having a deep understanding of these standard traditional ML models was important learning experience for us. We also provide evaluations for all these methods on our subset of BIGNEWSALGN dataset for test set.

## G. Evaluation Metrics

ML Model	Accuracy
LSTM	66.06
Keras NN	73.05
tensorflowNN	73.3
SVM	51.15
Naive Bayes	45.00

Fig 14 - Additional Work Evaluation Metrics

## X. CONCLUSION

In conclusion, our research introduces a sophisticated system that utilizes advanced Natural Language Processing techniques and Large Language Models to effectively classify and explain political biases in news media. This tool not only enhances the transparency of news reporting but also empowers users to critically engage with the content they consume, thus addressing the pervasive issue of echo chambers by providing nuanced insights into different perspectives. By integrating capabilities like query-based searches, direct text input, and URL analysis, our system offers a comprehensive approach to understanding media biases. Despite computational limitations, our models achieved strong performance benchmarks, demonstrating the potential of such technologies to foster a more informed and critically aware society. This work contributes significantly to the field of media studies, emphasizing the need for tools that support balanced news consumption and promote a healthier public discourse.

## REFERENCES

- [1] "Bias definition & meaning," Merriam-Webster, <https://www.merriam-webster.com/dictionary/bias#:~:text=bias%20implies%20an%20unreasoned%20and,against%20a%20person%20or%20thin g.> (accessed May 7, 2024).
- [2] N. Lee, Y. Bang, A. Madotto, and P. Fung, Mitigating Media Bias through Neutral Article Generation, <https://arxiv.org/pdf/2104.00336.pdf> (accessed May 7, 2024).
- [3] S. Aggarwal, T. Sinha, Y. Kukreti, and S. Shikhar, Media bias detection and bias short term Impact assessment, <https://www.sciencedirect.com/science/article/pii/S2590005620300102?via%3Dihub> (accessed May 7, 2024).
- [4] A. Radford et al., "Language models are unsupervised multitask learners - 2018," Language Models are Unsupervised Multitask Learners, [https://cdn.openai.com/better-language-models/language\\_models\\_are\\_unsupervised\\_multitask\\_learner s.pdf](https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learner s.pdf) (accessed May 7, 2024).
- [5] J. Wei et al., "Emergent abilities of large language models," arXiv.org, <https://arxiv.org/abs/2206.07682> (accessed May 7, 2024).
- [6] Y. Bang, D. Chen, N. Lee, and P. Fung, "Measuring political bias in large language models: What is said and how it is said," arXiv.org, <https://arxiv.org/abs/2403.18932> (accessed May 7, 2024).
- [7] T. B. Brown et al., "Language models are few-shot learners," arXiv.org, <https://arxiv.org/abs/2005.14165> (accessed May 7, 2024).
- [8] X. Sun et al., "Text classification via large language models," arXiv.org, <https://arxiv.org/abs/2305.08377> (accessed May 7, 2024).
- [9] Y. Zhang et al., "Teleclass: Taxonomy enrichment and LLM-enhanced hierarchical text classification with minimal supervision," arXiv.org, <https://arxiv.org/abs/2403.00165> (accessed May 7, 2024).
- [10] F. Wei et al., "Empirical Study of LLM Fine-Tuning for Text Classification in Legal Document Review," 2023 IEEE International Conference on Big Data (BigData), Sorrento, Italy, 2023, pp. 2786-2792, doi: 10.1109/BigData59044.2023.10386911.
- [11] A. Vaswani et al., "Attention is all you need," arXiv.org, <https://arxiv.org/abs/1706.03762> (accessed May 7, 2024).
- [12] H. Naveed et al., "A comprehensive overview of large language models," arXiv.org, <https://arxiv.org/abs/2307.06435> (accessed May 7, 2024).
- [13] Y. Liu et al., "Understanding llms: A comprehensive overview from training to inference," arXiv.org, <https://arxiv.org/abs/2401.02038> (accessed May 7, 2024).
- [14] H. Touvron et al., "Llama: Open and efficient foundation language models," arXiv.org, <https://arxiv.org/abs/2302.13971> (accessed May 7, 2024).
- [15] G. Team et al., "Gemma: Open models based on Gemini Research and Technology," arXiv.org, <https://arxiv.org/abs/2403.08295> (accessed May 7, 2024).
- [16] J. Bai et al., "Qwen Technical Report," arXiv.org, <https://arxiv.org/abs/2309.16609> (accessed May 7, 2024).
- [17] B. Zhang, Z. Liu, C. Cherry, and O. Firat, "When scaling meets LLM Finetuning: The Effect of data, model and finetuning method," arXiv.org, <https://arxiv.org/abs/2402.17193> (accessed May 7, 2024).
- [18] E. J. Hu et al., "Lora: Low-rank adaptation of large language models," arXiv.org, <https://arxiv.org/abs/2106.09685> (accessed May 7, 2024).
- [19] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "Qlora: Efficient finetuning of quantized llms," arXiv.org, <https://arxiv.org/abs/2305.14314> (accessed May 7, 2024).
- [20] R. Baly, G. D. S. Martino, J. Glass, and P. Nakov, "We can detect your bias: Predicting the political ideology of news articles," ACL Anthology, <https://aclanthology.org/2020.emnlp-main.404/> (accessed May 7, 2024).
- [21] R. R. R. Gangula, S. R. Duggenpudi, and R. Mamidi, "Detecting political bias in news articles using headline attention," ACL Anthology, <https://aclanthology.org/W19-4809/> (accessed May 7, 2024).
- [22] H. L. T. A. University et al., "Biaswatch: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management," ACM Conferences, <https://dl.acm.org/doi/abs/10.1145/2806416.2806573> (accessed May 7, 2024).
- [23] W.-F. Chen, K. A. Khatib, H. Wachsmuth, and B. Stein, "Analyzing political bias and unfairness in news articles at different levels of granularity," ACL Anthology,

- <https://aclanthology.org/2020.nlpccs-1.16/> (accessed May 7, 2024).
- [24] S. Raza, D. J. Reji, and C. Ding, “Dbias: Detecting biases and ensuring fairness in news articles,” arXiv.org, <https://arxiv.org/abs/2208.05777> (accessed May 7, 2024).
- [25] Y. Liu, X. F. Zhang, D. Wegsman, N. Beauchamp, and L. Wang, “Politics: Pretraining with same-story article comparison for ideology prediction and stance detection,” arXiv.org, <https://arxiv.org/abs/2205.00619> (accessed May 7, 2024).
- [26] L. Zheng et al., “Judging LLM-as-a-judge with MT-bench and Chatbot Arena,” arXiv.org, <https://arxiv.org/abs/2306.05685> (accessed May 7, 2024).
- [27] R. Taori et al., Alpaca: A Strong, Replicable Instruction-Following Model, <https://crfm.stanford.edu/2023/03/13/alpaca.html> (accessed May 7, 2024).
- [28] R. Jin et al., “A comprehensive evaluation of quantization strategies for large language models,” arXiv.org, <https://arxiv.org/abs/2402.16775> (accessed May 7, 2024).
- [29] M. Wessel et al., “Introducing MBIB -- the first media bias identification benchmark task and dataset collection,” arXiv.org, <https://arxiv.org/abs/2304.13148> (accessed May 7, 2024).