

ALLEGHENY COLLEGE
COMPUTER AND INFORMATION SCIENCE DEPARTMENT

Senior Thesis

**PolitScope; An Unsupervised
Natural Language Processing
Tool for Political Bias Detection**

by

Titus Smith

ALLEGHENY COLLEGE

**DEPARTMENT OF COMPUTER AND
INFORMATION SCIENCE**

Project Supervisor: **Emily Graber**
Co-Supervisor: **Douglas Luman**

Abstract

PolitScope is a web-based browser extension designed to address the growing issue of political bias present in social media, a challenge that contributes to the increasing polarization in society today. As bias escalates on social media, it's more important than ever to have reliable methods for detecting and analyzing it. This study explores the use of natural language processing (NLP) to create an automated system for identifying political bias. The methodology involves cleaning and embedding text using the RoBERTa model, then clustering similar text to create a baseline model. This model is saved and used to compare and dynamically detect bias in newly collected web text. Through a series of experiments, we evaluate PolitScope's effectiveness, examining how various parts of speech influence bias detection. While the results show that PolitScope is not yet perfect, the tool demonstrates the potential of NLP in highlighting political bias in online social media platforms, and contributing to the broader conversation about bias in digital media.

Table of contents

1	Introduction	6
1.1	Motivation	8
1.2	Current State of the Art	10
1.3	Assumptions, Limitations, and Delimitations	12
1.3.1	Assumptions	12
1.3.2	Limitations	12
1.3.3	Delimitations	13
1.4	Goals of the Project	13
1.5	Ethical Implications	14
2	Related work	16
2.1	Political Polarization and Media Bias	16
2.2	Social Media as a Platform for Political Discourse	17
2.3	Bias Detection and Sentiment Analysis Techniques	18
3	Methods	21
3.1	Introduction of PolitScope’s Development	21
3.2	Ethical Considerations in Development	21
3.3	High-Level Methodology	22
3.3.1	Model Development	22
3.3.2	Bias Detection and Clustering Analysis	23
3.3.3	Frontend	24
3.4	Preprocessing Data	24
3.5	Bias Detection	25
3.5.1	Embedding Generation	26
3.5.2	Model Development: Clustering Embeddings	28
3.5.3	Cluster Comparison	30
3.6	User Interface Integration	32
3.7	API Communication	34
3.8	Using PolitScope	35
3.9	Technical Stack and Library Usage	36
3.10	Challenges	37
4	Experiments	39
4.1	Experimental Design	39
4.1.1	Confidence Validation	39
4.1.2	Linguistic Impact	41
4.2	Evaluation	43
4.2.1	Confidence Validation Results	43
4.2.2	Linguistic Impact Results	45
4.3	Threats to Validity	46
4.4	Future Research Experiments	47

5	Conclusion	49
5.1	Summary of Results	49
5.2	Future Work	49
5.3	Future Ethical Implications and Recommendations	51

List of Figures

1	PolitScope Overview	7
2	Training and Comparison Pipeline	24
3	Softmax Function	27
4	3-Dimensional Visual Representation of Embedding Vectors from RoBERTa	29
5	3-Dimensional Visual Representation of PolitScope Cluster As- signment	31
6	Example PolitScope Output	36
7	Linguistic Impact Experiment Data	42
8	PolitScope's Accuracy Based on Confidence Coefficients	44
9	Subject, Adjective, and Nouns Impact on Sentence Embeddings .	45

List of Tables

1 Introduction

In an age dominated by digital media, the information we consume has a profound impact on our perceptions, beliefs, and decisions. However, much of this content is shaped by underlying biases that are often hidden from view. Bias is a human construct, that refers to a tendency, prejudice, or inclination that skews perception, judgment, or action in a particular direction, often in a way that is unfair or unbalanced. Bias can manifest consciously, where individuals are aware of their prejudices, or unconsciously, where they may act or think in biased ways without recognizing it. This is influenced by various factors, including cultural, social, and individual experiences. Online, biased content often emerges through algorithms, content prioritization, or selective exposure to information, affecting how individuals interpret or engage with digital content.

As individuals navigate their increasingly complex social media feeds, news websites, and online discussions, subtle biases in language and framing can influence their perspectives without their awareness. When we encounter unrecognized biases, they influence our perspective, or beliefs, and even how we act. This phenomenon is applicable in many fields of study, such as product consumption, cultural perceptions, and political views. Political bias contributes to the growing political polarization seen in the United States, creating an urgent need for tools that promote critical media consumption. Political polarization refers to the growing divide between ideological extremes, where individuals or groups are pushed toward opposing ends of the political spectrum, often leading to social and political division.

Our research is designed to meet this need. We developed a tool; PolitScope, that is a browser extension aimed at identifying political bias in real time as users browse the web. By leveraging advanced natural language processing (NLP) techniques and pre-trained models, PolitScope dynamically detects and highlights language that may include some varying level of underlying political bias. NLP is a branch of artificial intelligence that enables computers to understand, interpret, and analyze human language. PolitScope is designed to help users better understand the perspectives embedded in the content they consume. Unlike tools that require manual input or focus on fact-checking content, PolitScope operates seamlessly in the background, providing a user-free experience, and focused on the underlying bias that may not be as recognizable to an average media consumer.

To guide our research, we aim to address the following key questions:

- **RQ1** How effectively can a tool identify political bias in online content using NLP techniques?
- **RQ2** How can the use of a tool like PolitScope help analyze and characterize the political text online?

By exploring these questions, we hope to advance the understanding of political bias and provide users with the tools to make more informed decisions

about the media they engage with. Additionally we are aiming to develop a greater understanding of how political bias takes form in text; what patterns appear, what charged words hold more value, etc.

To uncover and classify political bias in online content, PolitScope employs advanced natural language processing (NLP) techniques and machine learning models. The foundation of this approach lies in RoBERTa (Robustly Optimized BERT Pretraining Apporoach), a pre-trained transformer model designed for robust textual analysis. By leveraging RoBERTa’s capabilities, PolitScope extracts word embeddings, which are mathematical representations of words and their relationships directly from textual data. These embeddings are able to create numeric representations of human text that represent the context, and meaning. It accomplishes this by recognizing nuanced patterns in tone, context, and word usage that are often indicative of bias.

After generating the embeddings, PolitScope uses a K-means clustering algorithms, and groups related words and phrases together into meaningful clusters that represent whether or not the text was classified as bias or unbiased. By conducting these analyses, the tool aims to successfully classify political bias with precision. Beyond classification, our research experiments using PolitScope enabled deeper evaluations of political text by exploring critical questions such as which words or phrases contribute most to bias, whether context has an impact, and how bias shifts in response to different textual inputs. This dual focus on detection and evaluation allowed PolitScope to be used as both a diagnostic tool and a research tool for advancing our understanding of political bias.

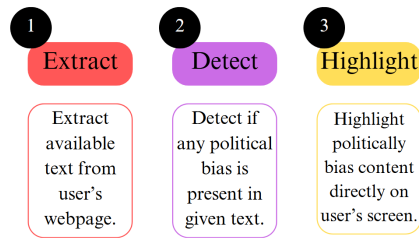


Figure 1: PolitScope Overview

Figure 1 illustrates the overall workflow of PolitScope, from extracting text on a webpage to identifying and highlighting political bias. The diagram provides a high-level overview of the three main steps: capturing the text content from a webpage, analyzing it for political bias using NLP techniques, and dynamically marking biased sentences directly within the user’s browser. This visualization helps clarify how PolitScope functions in real time, demonstrating how the tool directly integrates into the browsing experience to enhance users’ awareness of potential bias in online content.

The significance of this research extends beyond individual users. In today’s polarized political environment, the tool seeks to foster greater media literacy,

encouraging critical evaluation of the information we encounter daily. Media literacy involves the ability to access, analyze, evaluate, and create media in various forms, enabling individuals to understand the impact of media on society. By uncovering bias and promoting awareness, PolitScope has the potential to empower individuals to make more informed decisions and contribute to healthier public discourse. This is a first (yet crucial) step towards addressing the diverging political polarization that plagues the United States.

Overall, it is important to understand why a tool like PolitScope is not just helpful, but necessary for navigating the complex and often biased landscape of political content online. In today’s digital landscape, the way we consume news has shifted dramatically, with social media becoming a primary source of information, especially for younger generations [13]. However, this shift comes with its own set of challenges. Unlike traditional media, which is subject to more rigorous editorial standards, digital platforms are filled with content that often carries hidden political biases. These biases can subtly influence the way we think, act, and perceive the world around us.

Political content on social media is particularly powerful because it is designed to persuade, influence, and convince. This type of content doesn’t just inform, it shapes opinions, often in ways we don’t realize. When left unchecked, these biases contribute to growing political polarization, creating an “us versus them” mentality that divides society further. This polarization has profound consequences for public discourse and democracy, making it harder to find common ground and fostering an increasingly adversarial political environment.

While awareness of misinformation has grown, with tools now available to help users identify false or misleading information, these tools fall short when it comes to addressing bias [7]. The problem of bias is more nuanced and insidious than mere misinformation. Misinformation is the false or inaccurate spread of information, regardless of intent. Bias doesn’t just change what we believe; it shapes how we think, often without us even knowing it. This lack of awareness leaves users vulnerable to manipulation, with their thoughts and beliefs subtly molded by the content they consume.

PolitScope seeks to address this critical gap. Unlike existing tools, which focus on verifying truth versus falsehood, PolitScope is designed to identify and expose the underlying biases hidden within political content. By bringing these biases to light, we can empower individuals to critically evaluate the information they encounter and take the first step toward addressing the larger issue of political polarization in the United States.

1.1 Motivation

In today’s digital era, the rise of social media as a dominant source of information has significantly transformed how individuals, particularly younger generations, engage with the world. Platforms like Twitter, Instagram, TikTok, and Facebook now serve as primary platforms for news consumption and political discourse. According to recent studies, young people increasingly rely on social media for political content, with nearly half of individuals aged 18 to 29

reporting that it is their main source of news [13]. This trend raises serious concerns about the quality and impact of the information consumed. Unlike traditional media outlets, which are held to editorial standards, social media content operates in a largely unregulated environment where misinformation, bias, and sensationalism flourish.

Politicians and political organizations have recognized the power of social media, investing unprecedented amounts of time and money into producing targeted content designed to sway public opinion. Campaign spending on digital advertising has skyrocketed in recent election cycles, with billions of dollars funneled into social media platforms to create highly tailored messages that target and influence specific demographics [2]. This focus has made political content on social media a pervasive presence in the lives of young users. However, the mechanisms driving this content are not designed with the user’s best interests in mind. Algorithms prioritize engagement above all else, tailoring feeds to show content that elicits strong emotional responses, often through polarizing or biased material. This emphasis on engagement perpetuates echo chambers, where individuals are exposed predominantly to views that align with their own; a phenomenon known as confirmation bias [8].

Confirmation bias plays a critical role in shaping perceptions and beliefs. When users encounter information that aligns with their existing viewpoints, they are more likely to accept it without question, reinforcing preexisting attitudes and closing off opportunities for balanced, critical engagement. This dynamic is particularly troubling for younger individuals, who are still in the process of developing their beliefs and understanding of the world. Consistent exposure to biased or one-sided content during formative years can deeply influence their worldview, often without their awareness. As a result, social media becomes a powerful force in entrenching political divisions and contributing to the growth in polarization seen and felt in the United States today [31] [6].

Social media platforms, driven by their design to maximize user engagement, do little to address this issue. Their algorithms are not concerned with the quality or intent of the content they promote, prioritizing clicks, shares, and watch time over accuracy, balance, or the public good. This creates an environment where biased, inflammatory, or even misleading political content thrives, not because it informs or educates, but because it keeps users scrolling. The lack of safeguards against the subtle manipulation of beliefs highlights the need for tools that empower individuals to critically evaluate the content they encounter.

This research is motivated by the urgency of addressing these issues. The growing polarization in the United States is not merely a byproduct of political disagreements; it is fueled by the media we consume and the unchecked biases embedded within it [17]. Social media has become a battleground where individuals’ beliefs and identities are shaped, often without their conscious awareness. It is essential to intervene in this process, particularly for young audiences who are most vulnerable to the influence of digital content.

The development of PolitScope represents a step toward mitigating this problem. By providing a tool that identifies political bias in real time, users can engage more critically with the content they consume. PolitScope’s focus on

accessibility and smooth integration into daily browsing ensures it can reach a broad audience, equipping them with the tools needed to recognize and question bias. This, in turn, fosters more informed and balanced discourse; a necessary foundation for addressing the polarization that threatens democratic society. By enabling individuals to better understand the forces shaping their perspectives, PolitScope aims to create a more aware, critically engaged populace, which is the first step toward minimizing the divisive effects of political polarization.

1.2 Current State of the Art

Existing tools for analyzing political content online typically fall into two categories: misinformation detection and sentiment analysis. In recent years, platforms like Twitter and Instagram have begun to include a misinformation flag for posts. However, misinformation detection tools often rely on binary true/false flags to classify content as accurate or false [30] [7]. While these tools are effective for addressing fact-based inaccuracies, they fail to capture the subtleties of political bias, which operates in a more nuanced spectrum rather than binary. Political bias in content is far more nuanced than a binary classification can capture. It can manifest in the tone, framing, or context of the language used, and these subtleties are often overlooked by current tools. While not in the current version of PolitScope, an addition to this tool could explore representing the strength or severity of bias using visual cues.

The limitations of existing tools highlight a significant gap in the ability to understand and analyze political bias in everyday online content. While tools focused on misinformation are useful, they miss the broader issue of political bias, which shapes how people think and act, often without their awareness. For example, the subtle framing of an issue in a news article or a social media post can influence opinions, even when the factual accuracy of the content is not in question. Current tools fail to address this problem comprehensively.

In addition to these tools, sentiment analysis has become a popular method for examining political content, particularly in academic research. This approach focuses on evaluating the emotions expressed within a text (such as anger, joy, or fear) and categorizing these emotions based on political alignment. However, while this research is valuable, it often remains confined to academic settings and is not readily accessible to the average media consumer. In a time when political polarization is rampant, there is an urgent need to bridge this gap and make such insights available to the public, especially in a format that doesn't require specialized knowledge or academic engagement.

To overcome these limitations, PolitScope seeks to integrate political bias detection into a single, easy-to-use tool that is accessible to everyday users. Unlike tools that require users to manually input content for analysis, PolitScope operates seamlessly in the background, dynamically identifying and highlighting political bias as users interact with online content. By making these tools available to the average user, PolitScope aims to empower individuals to critically assess the political content they encounter and become more informed consumers of digital media.

Political bias detection is inherently challenging due to the complexity of bias itself. Bias can be subtle, context-dependent, and shaped by cultural, personal, or regional factors. What one person perceives as biased, another might see as neutral or justified. Adding to this complexity, bias is not merely a matter of word choice or tone; it often emerges from the interplay between language, context, and framing. Existing tools fail to account for these intricacies, relying on limited case studies, such as a specified time period or geographical location, labels, or datasets that do not reflect the diverse perspectives inherent in real-world discourse [35].

To address these gaps, our research leverages advancements in Natural Language Processing (NLP), which has revolutionized the analysis of text in recent years. Early NLP approaches were limited in their ability to understand the complexities of human language, focusing on simple keyword matching or basic statistical patterns. However, the introduction of transformer-based models like RoBERTa has enabled NLP to analyze text with unprecedented depth and accuracy. These models excel at capturing context, tone, and relationships between words, which serve as key elements in understanding political bias.

RoBERTa was chosen as the foundation for PolitScope due to its ability to process text with high precision, making it well-suited for detecting bias in complex and context-rich political content. The tool employs RoBERTa to generate word embeddings, mathematical representations of language that capture both semantic meaning and relationships between words. Then, by applying clustering algorithms to these embeddings, PolitScope identifies patterns and subtle biases that might escape the notice of a human observer.

A key reason we chose to rely on NLP for our classification comes from the inherent subjectivity involved in manually detecting bias. Human analysis is prone to personal biases and inconsistencies, which can distort findings and reduce trust in the results. By using machine learning models trained on diverse datasets, PolitScope minimizes these risks, offering an objective, data-driven approach to bias detection. This objectivity allows PolitScope to act as a neutral intermediary, helping users critically evaluate content without imposing an external agenda.

Despite these advancements, the field of NLP faces its own challenges, such as understanding cultural nuances, dealing with ambiguous or sarcastic language, and adapting to the rapidly changing landscape of digital media. While not directly tested, the integration of clustering, in addition to the NLP analysis, aims to elevate PolitScope to accurately address these limitations.

In summary, while the current state of the art provides valuable tools for misinformation detection and sentiment analysis, it falls short in addressing the subtleties of political bias in a way that is both accessible and actionable for the average user. By leveraging the latest advancements in NLP and addressing the inherent subjectivity of human analysis, PolitScope offers an unsupervised, data-driven solution for identifying and understanding political bias in digital media.

1.3 Assumptions, Limitations, and Delimitations

1.3.1 Assumptions

In developing PolitScope, several assumptions were made to guide the analysis of political bias on social media platforms. One key assumption is that language on social media follows identifiable trends and patterns. This includes the frequent use of hashtags, emoticons, political terminology, and phrases that can be consistently detected by natural language processing (NLP) tools. It is assumed that political posts, particularly on platforms like Twitter, exhibit discernible patterns of sentiment, bias, or polarity, which machine learning models can capture effectively.

Additionally, it is assumed that the machine learning models used in this research, such as RoBERTa and K-means clustering, are capable of being generalized across different social media platforms. These models are expected to detect patterns in political discourse regardless of platform-specific variations in user behavior or content format. This assumption is based on the premise that context, meaning, and bias can be captured consistently across diverse social media contexts.

1.3.2 Limitations

Despite the assumptions, there are several limitations inherent in this approach. First, there is the challenge of data availability and representativeness. Social media data might not fully capture the political spectrum or be representative of the broader population. Due to privacy settings and API restrictions, large-scale data collection, especially from platforms like Twitter, may face significant obstacles.

Another limitation lies in the subjectivity of political bias. Political bias is often context-dependent and subjective, which makes it difficult to label with certainty. Different users may interpret content differently, and the same text might evoke opposing sentiments based on the individual’s political viewpoint. This variability can lead to some disagreeing with the results of our model, especially when detecting subtle, implicit, or indirect forms of bias. Importantly, PolitScope aims to address this limitation directly to users by encouraging users’ to conduct further research of the results.

The informal nature of social media language also presents challenges. Social media users often employ slang, abbreviations, emojis, and memes, which are difficult for NLP models to interpret accurately. These variations may affect the precision of the analysis, as models trained on more formal language may struggle with the non-standard communication format commonly found on social media.

A significant limitation is that the models used in PolitScope, including RoBERTa, might not detect all forms of bias, particularly those that are subtle or implicit. Biases that are deeply ingrained or expressed indirectly through language may evade detection. Moreover, relying on pre-trained models such as RoBERTa means the system is constrained by the models’ limitations and

their ability to accurately capture and interpret political bias in a fast-evolving domain like politics. Unexpected shifts in political events or public opinion can rapidly alter the political landscape, making models trained on outdated data less effective.

1.3.3 Delimitations

To ensure focus and feasibility, certain delimitations were established for PolitScope. One of the primary delimitations is the choice of platform for analysis. The tool is designed for text-based analysis, and social media platforms like Instagram and TikTok, which rely heavily on images and videos, fall outside the scope of this research. Consequently, this tool may not be as useful for detecting political bias on those platforms.

The project also delimits its scope to political bias detection using specific pre-trained models such as RoBERTa. While there are other machine learning techniques and models available, these tools are selected for their relevance to the project’s goals, and other methodologies are excluded from consideration in this phase of research.

Finally, politics is inherently fluid, and PolitScope acknowledges that large-scale political changes or shifts in public opinion may require updates to the models and the data they are trained on. If the political landscape evolves significantly, the model will need to be retrained and refined to ensure continued accuracy and relevance.

1.4 Goals of the Project

The primary objective of this project is to address our two research questions: **RQ1:** How effectively can a tool identify political bias in online content using NLP techniques, and **RQ2:** How can the use of a tool like PolitScope help analyze and characterize the presence of political bias in online text? These questions are not just academic, rather they represent a critical step toward building a future in which political engagement is less driven by unrecognized bias and polarization. By equipping users with the ability to identify bias in the content they consume, we aim to empower them to critically evaluate information and reduce the effects of confirmation bias. Aiming to ultimately contribute to a more informed and balanced public discourse.

At its core, the goal of PolitScope is to foster awareness of bias. Awareness is the first step in addressing a problem as pervasive and insidious as political polarization. Many users remain unaware of the subtle ways in which biased language and framing shape their perceptions, reinforcing existing beliefs and deepening divisions. Our research seeks to change this by highlighting bias in real time, offering users a tangible way to see how language influences their understanding of political content. This recognition is a necessary precursor to meaningful action; only when individuals recognize that bias exists can they begin to critically engage with it and work toward limiting its effects.

This tool is not designed to offer solutions to political polarization overnight. Instead, it is a starting point, or a means of initiating a broader conversation about how we consume and interpret political content in the vast world on online content. By providing users with a clear, accessible way to see bias in the media they encounter, PolitScope is laying the groundwork for a larger cultural shift. The long-term goal is to reduce the impact of confirmation bias, encourage exposure to diverse perspectives, and foster healthier, more constructive political dialogue.

In addition to its immediate impact on individual users, PolitScope represents the first step in a larger process of combating political polarization at scale. Recognition is a foundational step, but it must be followed by education, dialogue, and systemic change. While our tool’s primary function is to detect and highlight bias, its broader purpose is to serve as a stepping stone for the following steps. By spreading awareness and equipping individuals with the tools to recognize bias, the project is contributing to a cultural shift that prioritizes critical thinking and media literacy.

Ultimately, the goal of this project extends beyond the technical achievement of developing an effective NLP-based bias detection tool. It seeks to address a societal issue by empowering individuals to take control of their media consumption and engage with political content in a way that is thoughtful, informed, and less prone to the pitfalls of polarization. By bridging the gap between academic research and everyday media use, PolitScope aspires to be a tool not just for analysis, but a significant step toward a less polarized future.

1.5 Ethical Implications

The development of our research raises ethical considerations that are crucial to understand, particularly regarding the collection, analysis, and interpretation of data. Since this project utilizes publicly available tweets and text content from public emails, it is important to address privacy concerns to ensure that the tool operates responsibly [20] [14]. While tweets are inherently public, the ethical principle of respecting user anonymity is an important consideration. By using deidentified data, we minimize the risk of exposing individual users’ identities, ensuring that no personal details are linked to the data being analyzed. Beyond anonymization, obtaining informed consent from users remains a challenge due to the nature of publicly available data. Although explicit consent may not be feasible for every user, adherence to ethical guidelines for public data usage is essential to strike a balance between research objectives and respect for individual rights. Overall, to ensure user privacy, all data used to train PolitScope was fully anonymized, preventing any connection to identifiable individuals.

Transparency also plays a critical role in the ethical implementation of a tool like PolitScope. Clearly communicating the methods and algorithms used in its development fosters trust and understanding among users. Openly sharing details about how the tool detects bias, its underlying assumptions, and the limitations of its model is essential. This transparency not only helps limit skepticism but also encourages users to critically evaluate the results PolitScope

provides, understanding it as a tool for insight, and may always give fully accurate results.

One of the central ethical concerns is the inherent bias that may exist within the tool itself. PolitScope is built using natural language processing (NLP) models and machine learning algorithms, both of which can inadvertently reflect biases present in their training data or design. For example, if training data disproportionately represents particular viewpoints or linguistic patterns, the tool may overemphasize certain biases while under-detecting others. This challenge highlights the importance of testing, refinement of the algorithm(s), and transparency about the tool’s limitations. While PolitScope strives to reduce subjective bias in detecting political content, no system is entirely free from the influence of the human assumptions and decisions made during its creation.

Algorithmic bias is another critical concern. Machine learning models, like those used in PolitScope, operate based on patterns detected in training data, which can lead to systemic errors when applied to diverse real-world contexts. For instance, cultural nuances, linguistic variations, or evolving political discourse may not be adequately captured, resulting in misclassifications or oversimplifications. To mitigate this, the tool’s development is built on a diverse datasets of tweets and news/blog text, and ensures its output reflects an understanding of political bias.

Additionally, the very nature of bias introduces ethical complexities. Users may rely on the tool to confirm pre-existing notions or to validate their perspectives, potentially adding to confirmation bias rather than combatting it. This risk shows the importance for careful framing in the tool’s implementation. PolitScope aims to highlight bias to promote critical thinking, but without appropriate guidance, users may misuse its findings. To address this, the tool emphasizes informed use, providing clear explanations and encouraging users to engage with content thoughtfully.

Finally, the broader societal impact of our research must be considered. While the tool is designed to combat political polarization and promote media literacy, its potential misuse cannot be ignored. Users could exploit its insights to manipulate narratives, reinforce misinformation, or target specific groups. Additionally, political bias is deeply intertwined with cultural and historical contexts, and a tool designed primarily for U.S. political discourse may not translate effectively to other regions without significant adaptation. Recognizing these challenges, our research is positioned as an educational tool rather than a definitive solution, with its results framed as a starting point for deeper reflection and analysis.

By addressing privacy, transparency, bias, and the potential for misuse, our research seeks to navigate the ethical complexities inherent in its mission. It is crucial to consider these ethical dilemmas to ensure that our tool not only meets the goals but is also fair, respectful, and ethical. The following chapters will explore related literature that informs our research, detail the methods used to develop PolitScope, present experiments designed to test and validate its effectiveness, and conclude with a discussion of potential future improvements to the tool.

2 Related work

This chapter provides an overall review of the existing literature and research related to the development and motivation for our research: PolitScope. Our goal is to contextualize the current state of knowledge in the areas of political polarization, media bias, social media’s role in shaping political discourse, and the technical methodologies used to detect bias in text. Each of the following sections highlight key research that informs and motivates the design and objectives of this project. We will explore various approaches to sentiment analysis in the political field, machine learning techniques, and the challenges of data collection and preprocessing in the political domain. By summarizing the current state of work, we aim to identify gaps, contextualize the motivation, and explain the foundation that built PolitScope. This research sets the stage that led us to develop a tool that addresses political polarization and media bias through real-time detection and user engagement.

2.1 Political Polarization and Media Bias

Political polarization in the United States has reached unprecedented levels in recent decades, becoming a central issue that affects the functioning of democracy. Studies reveal that the U.S. exhibits the fastest-growing polarization of any democracy worldwide, highlighting a widening ideological divide between the left and right [33] [17] [31]. This increasing polarization is not only a political issue but one that shapes our society as a whole, undermining institutions, and fracturing public discourse. Research highlights the severity of this trend, emphasizing how political polarization damages democracy by weakening the independence of the judiciary, fostering legislative gridlock, and even harming societal norms [3] [11]. The polarization of political opinions has resulted in a climate where tolerance and moderation are becoming increasingly rare, causing divisions in what were once considered common social bonds.

One contributing factor to the deepening polarization is the pervasive influence of social media. Research indicates that people aged 19-29, for instance, spend over 2.5 hours per day on social media platforms [15]. These platforms serve as key spaces for political discourse, but they often amplify existing biases, promoting echo chambers that reinforce users’ pre-existing beliefs. This has profound implications, as individuals are exposed to information that aligns with their views while avoiding dissenting perspectives. Moreover, the increasing role of political ads on social media, driven by substantial financial investments, contributes to an environment where political messages are not only targeted but also tailored to further polarize the audience.

Compounding the issue of political polarization is the role of media bias. Studies point out that biased media outlets, while often criticized, are incredibly profitable, with high-bias news organizations outperforming low-bias ones in terms of profitability [11]. This profit-driven model incentivizes the propagation of sensationalist, often divisive, narratives that attract viewers but deepen the ideological divides. These biases are not just evident in traditional media outlets

but also pervade the digital sphere, where algorithms favor content that aligns with a user’s preferences, further entrenching polarization. The question then becomes: How do we address the consequences of these biases, especially when bias is so ingrained in the platforms that most people engage with daily?

The dangers of polarization are not just theoretical; they manifest in tangible, harmful ways. Previous research shows that polarization weakens social cohesion and undermines democratic practices [24]. A polarized public, divided along ideological lines, is less likely to engage in constructive discourse or seek common ground, which is essential for problem-solving in a democratic society. This fragmentation, as emphasized in research, is not just a passing trend but an evolving pattern that increasingly shapes the political landscape [17].

In conclusion, these works not only highlight the deepening issue of political polarization and media bias but also frame the necessity for solutions like PolitScope. They set the stage for understanding the critical need for tools that provide both immediate feedback and long-term solutions to the growing problem of biased political content, particularly in the ever-changing landscape of social media.

2.2 Social Media as a Platform for Political Discourse

Social media platforms have become the dominant space for political discourse in the modern era, influencing not only how individuals engage with politics but also shaping the nature of political debate itself. As mentioned before, research indicates that people spend significant portions of their day on these platforms [15] [5]. This high level of engagement has made social media a primary space for political content, with at least 40% of users reporting that some of the content they encounter is politically related [13]. As social media usage increases, political discussions increasingly occur in these digital spaces, making them central to the way people form and express political opinions.

The growing political presence on social media is not just a result of user engagement but is also actively driven by political campaigns. Politicians and interest groups have recognized the power of social media in shaping public opinion and have accordingly increased their investments in digital advertising. Statista’s research shows a marked rise in political ad spending on social media platforms, reflecting a broader shift toward these platforms as key tools in political communication [2]. This change has implications not only for the volume of political content users encounter but also for how such content is created and targeted to individuals based on their behavior and preferences.

However, this targeting and creation are not neutral processes. The algorithms that drive social media platforms are designed to engage users by delivering content that aligns with their existing views and interactions. A former Google design ethicist compared these algorithms to “voodoo dolls” in their effectiveness at controlling user behavior, highlighting how the platforms exert subtle but powerful influence over what users see [28]. These algorithms promote content that is more likely to elicit a strong emotional response, thus increasing user engagement.

At the core of these algorithm designs is a well-documented psychological phenomenon known as “negativity bias,” the tendency for individuals to pay more attention to and react more strongly to negative information. According to research from The Decision Lab, social media algorithms are built to take advantage of this bias, feeding users content that reinforces their negative emotions, which in turn amplifies polarization [26]. As negativity bias drives users to engage more deeply with emotionally charged, often divisive content, the platform’s algorithm ensures that such content remains at the forefront of their feeds, leading to deeper entrenchment in their ideological views. This dynamic further exacerbates political polarization, as individuals are not only isolated within their ideological bubbles but are also more likely to encounter content that confirms and intensifies their existing biases.

The result of these interconnected dynamics is a feedback loop where social media platforms actively contribute to the political polarization that is already widespread in society. By encouraging engagement through emotionally charged, often divisive content, social media becomes a platform that both reflects and magnifies the ideological divides present in the public sphere. The rise of digital political ads, combined with algorithm-driven content curation, creates a landscape where political discourse is increasingly characterized by echo chambers and hyperpolarization.

In conclusion, social media’s role in political discourse is not simply passive, it is a force that shapes political attitudes and behaviors. As social media platforms continue to prioritize engagement over content accuracy or diversity, they become breeding grounds for political polarization. Given how these platforms operate and the algorithms that underpin them, PolitScope is taking action to address the larger issues of political polarization and bias present in today’s digital age.

2.3 Bias Detection and Sentiment Analysis Techniques

The detection of bias in political discourse requires advanced techniques that can process, analyze, and classify large volumes of text. To address the challenges associated with identifying bias, a variety of machine learning models and natural language processing (NLP) methods have been explored in recent years. This section examines the effectiveness of different approaches, justifies the selection of specific techniques for the current project, and highlights the importance of using advanced models such as RoBERTa, and clustering algorithms for detecting bias.

One popular approach to unsupervised learning, useful for identifying patterns in unstructured text data, involves clustering methods such as K-means, and Latent Dirichlet Allocation (LDA)[21] [1]. These methods group data points based on similarity without requiring labeled data, making them ideal for exploratory analysis of data that may or may not contain political bias. K-means clustering, for example, organizes data into a specified number of clusters, after which, researchers can identify thematic similarities [9]. These techniques provide a starting point for exploring bias in text by categorizing related political

content, revealing underlying biases and patterns. While clustering methods are often used for discovering hidden patterns, they do not inherently account for the polarity of sentiments or the nuanced bias present in the content. This is where more sophisticated models come into play.

For more complex and context-aware analyses of bias, models such as BERT (Bidirectional Encoder Representations from Transformers) and RoBERTa (Robustly optimized BERT) have emerged as powerful tools in NLP [16] [19]. BERT, developed by Google, revolutionized sentiment analysis and bias detection by pre-training a deep learning model on large corpora of text data to predict missing words and understanding sentence context from both directions (left-to-right and right-to-left). This bidirectional approach enables BERT to capture contextual nuances that are crucial for detecting subtle biases in political discourse.

RoBERTa, an optimized version of BERT, improves on the original model by training on even larger datasets and removing certain training constraints, allowing it to better handle a broader range of language use. This model has achieved state-of-the-art performance on a variety of NLP tasks, including sentiment analysis and bias detection [19] [22]. Our use of RoBERTa in this project is grounded in its proven ability to understand complex linguistic features and identify both explicit and implicit biases within text.

In contrast to using RoBERTa, sentiment analysis has long been a critical tool for understanding the emotional tone behind political messages. Historical research on sentiment analysis in political discourse has demonstrated the utility of machine learning techniques like Long Short-Term Memory (LSTM) networks and traditional classifiers (e.g., Support Vector Machines, Decision Trees) in analyzing sentiments expressed in political communications [10].

While traditional sentiment analysis methods can offer insight into whether content is positive or negative, they are less equipped to capture more subtle biases. This is why the use of advanced NLP models like RoBERTa are crucial, as they may be able to represent not only the sentiment but also the underlying bias, particularly in cases where the bias is embedded in more complex linguistic structures, using NLP vectorization. NLP vectorization is the process of converting human language into numerical representations, and plays a key role, as it allows researchers to detect and learn from intricate linguistic patterns that may convey bias [27]. Therefore, the integration of RoBERTa and clustering methods on political text provides a more comprehensive approach to detecting bias, capturing both the emotional tone and the underlying bias present in political discourse.

The combination of advanced clustering techniques, text embedding models, and sophisticated NLP architectures like RoBERTa provides a robust framework for detecting bias in political texts. By utilizing these models, this project aims to enhance the accuracy of bias detection, providing important insights into political polarization and its harmful effects. The lessons learned from previous works, such as the application of LSTM and traditional classifiers in sentiment analysis and the use of unsupervised learning for clustering political texts, guided our decision to utilize the RoBERTa model, and K-means clus-

tering techniques to build a data-driven tool. This multi-layered approach is intended to create a strong analysis of bias, and create an accurate tool, which will lead to less polarized political discourse.

3 Methods

3.1 Introduction of PolitScope’s Development

We developed PolitScope as a tool to analyze political bias in textual content using unsupervised NLP and machine learning techniques. We designed the project with the goal in mind to provide users with an objective measure of bias within political discourse by leveraging clustering algorithms, and an embedding-based NLP analysis. The development process involves several key stages: data collection, text preprocessing and cleaning, embedding generation, clustering analysis, and visualization.

At its core, this process aims to bridge the gap between raw political content and structured analytical insights. By processing multiple forms of textual content, from a variety of political boards, the tool provides a way to assess bias trends. Throughout the development process, ethical considerations were extremely important to ensure that the analysis did not introduce additional/personal biases or misrepresent political perspectives. Importantly, the goal was not to label content as inherently “good” or “bad” but rather to simply highlight patterns in language that might help indicate political bias.

3.2 Ethical Considerations in Development

One of the biggest challenges in building PolitScope was handling the inherent subjectivity of political discourse. Since bias is often present in subtle word choices, framing, or omitted information, it was crucial to ensure that our model did not reinforce existing biases present in the training data. We addressed these concerns by implementing diversity measures without our data collection, ensuring balanced representation of political viewpoints during model training.

Additionally, user privacy was a major concern. Unlike other NLP-based analytics tools, PolitScope does not store user input or use personal data for analysis. Our system processes text in real-time without retaining any information beyond the immediate session. This approach ensures that users can analyze content without concerns about data tracking or privacy violations. Additionally, it is important that we get informed consent from all users using PolitScope to gain access to the text on their screen, ensuring clear and transparent data collection from the user, on top of transparent measures on what happens with their data.

In our implementation, we utilize RoBERTa, a pre-trained transformer-based NLP model. While RoBERTa is a powerful tool for natural language processing tasks, it is important to acknowledge that it was trained on large-scale datasets that may contain implicit biases. These biases can emerge from the original data sources and influence how the model processes and interprets language. This is a well-documented challenge in AI ethics, as even models trained on diverse datasets can reflect and perpetuate societal biases. Recognizing this limitation, we remain cautious about the results generated by RoBERTa and provide warnings to the users of the possibility that our results may contain

inherent bias due to the use of NLP techniques.

Furthermore, while we have implemented several bias mitigation techniques as recommended by best practices in algorithmic fairness, bias and correctness often exist in a complex, intertwined relationship. Addressing bias does not merely involve removing certain flagged words or phrases, as this could distort meaning or introduce unintended consequences. Instead, bias mitigation requires a holistic approach that includes dataset curation, model fine-tuning, human oversight, and continuous re-evaluation of results [25]. Even with these measures in place, no automated system can guarantee absolute neutrality, especially in politically charged discourse.

To ensure transparency and prevent potential misinformation, we have embedded an important notice within the extension itself. This notice informs users that our tool relies on NLP techniques, which, while sophisticated, are not infallible. It encourages users to approach the tool’s findings critically and conduct additional research for further clarity. The disclaimer is placed in close proximity to the flagged content, ensuring that users immediately associate the warning with the information being highlighted [32]. By aligning the notice with best practices in user interface design, we reinforce the connection between the flagged text and the potential for algorithmic error, reducing the likelihood of misinterpretation. This disclaimer serves as a safeguard against over-reliance on the tool and reinforces the idea that algorithmic outputs should complement, rather than replace, independent analysis and judgment. By incorporating this level of transparency, we strive to create a tool that genuinely aids users in navigating media bias rather than unintentionally reinforcing pre-existing prejudices.

3.3 High-Level Methodology

The development of PolitScope is structured into several interconnected components, each playing a vital role in the overall analysis pipeline. The process we chose can mainly be broken down into two main subsections, the backend process which handles model development (3.3.1) and bias detection of user text (3.3.2), and the front end process handling data collection and reporting results (3.3.3).

3.3.1 Model Development

The following sections outline the key steps involved in the training and development of PolitScope’s model. Each category represents a crucial component in building the system, from preprocessing text data to implementing the bias detection algorithm. The backend of PolitScope was built using Python, which provided flexibility and access to a wide range of natural language processing libraries.

Data Handling and Preprocessing

- The project processes textual data primarily sourced from social media,

ensuring that informal language, slang, and hashtags are preserved for accurate bias detection.

- Preprocessing involves sentence segmentation, lowercasing, and punctuation removal while maintaining the integrity of hashtags, slang, and other stylistic elements relevant to political discourse.

Embedding Generation and Model Training

- In a one-time training step, we use the transformer-based NLP model RoBERTa to convert a large, labeled dataset into dense numerical representations known as embeddings.
- The embeddings capture contextual meaning and relationships between words, forming the basis for nuanced analysis of political language.
- We then apply a clustering algorithm to group the embeddings based on numeric proximity, using Euclidean distances, because text with similar patterns across the dataset should have similar embeddings.
- Entropy-based confidence scoring is also calculated for each embedding to quantify the model’s certainty about that the numeric embedding accurately reflects the data it is representing.
- The resulting model, consisting of cluster centroids and entropy thresholds, is saved for future comparison and classification.

3.3.2 Bias Detection and Clustering Analysis

- In the iterative process, user-provided or web-scraped text is cleaned and embedded using the same RoBERTa model as in training.
- Each new embedding is compared to the pre-trained cluster centroids to assess its ideological alignment.
- Both the distance to the closest cluster and the entropy score are used to determine whether the text is confidently biased.
- This repeatable process allows PolitScope to classify new text inputs against the trained model, enabling scalable and efficient bias detection.

The relationship between the one-time training step and the iterative bias detection process forms the foundation of PolitScope’s analysis pipeline. By separating model creation from real-time classification, we ensure both efficiency and consistency in detecting bias. This relationship is illustrated in the Figure 2, which maps the flow from model training to ongoing text evaluation.

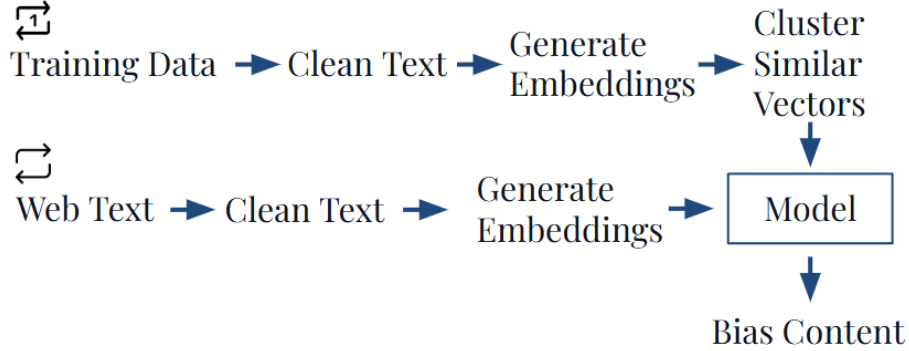


Figure 2: Training and Comparison Pipeline

3.3.3 Frontend

The following sections detail the essential steps in developing the front end of PolitScope. These components extract of webpage content, communicate with the back end for bias detection, and effectively present results directly within the user’s browsing experience.

User Interface and Integration

- The frontend was built using JavaScript and HTML, providing an accessible interface for users to input text and receive bias assessments in real-time.
- The frontend is designed to extract textual information from a users webpage, and then return a list of potentially bias sentences, as well as directly highlighting that content on their screen.

By following this structured approach, PolitScope uses RoBERTa for the practical application aimed at increasing awareness of political bias in digital communication. The following sections will explore each component in greater detail, outlining the technical methodologies and design choices that shaped PolitScope’s development.

3.4 Preprocessing Data

To understand our development of the preprocessing phase of PolitScope, we must first anticipate the intended data our tool will likely be fed. Considering we designed PolitScope as a tool for the general public to use while browsing online content, including social media, it was important to develop our tool to handle textual data from informal settings such as social media and non-academic webpages. With this in mind, we aimed to clean social media text while preserving key linguistic elements that might contribute to bias detection. Given the nature of social media, where users frequently use hashtags, slang, abbreviations, etc., we designed our preprocessing step to focus on structural

cleaning rather than excessive normalization. Excessive normalization refers to the process of overly simplifying or standardizing text to the point where important linguistic variation and context are lost, reducing the richness of the original input. Specifically, our approach ensures that the integrity of the user-inputted text remains intact while removing noise that could hinder our analysis, or disturb our NLP model. Focusing on sentence segmentation and minimal text cleaning, allows our NLP model to capture the nuances of social media language, such as non-standard grammar, informal expressions, and rhetorical markers. As part of this process, we also discard sentences that are too short, as they often lack the context necessary for effective bias detection and can introduce noise into the model. This is an important step because excessive preprocessing could strip the text of valuable contextual and stylistic elements contributing to bias detection.

First in the implementation of our preprocessing step, we segment the text into individual sentences using a regular expression based sentence splitter that detects punctuation-based sentence endings (i.e., ., !, and ?). This was a crucial step in ensuring the textual data being fed to RoBERTa was in the correct format to get the best out of RoBERTa’s implementation. The specifics of RoBERTa’s implementation is explained in greater detail in the embedding section (3.5.1). Next, we perform a basic cleaning step that converts all text to lowercase and removes punctuation while preserving spaces and alphanumeric characters. This step ensures consistency across different input styles while avoiding too much data loss. Unlike many NLPs, we deliberately do not preprocess hashtags or slang terms because they often carry contextual weight, influencing the overall sentiment and political tone of the text. For example, the hashtag “#FakeNews” conveys a strong political connotation and should not be discarded or removed, as doing so could weaken the effectiveness of our bias classification model when analyzing textual content on social media.

Although our original plan included tokenization using RoBERTa’s AutoTokenizer, we ultimately opted to exclude this approach from our preprocessing implementation. Tokenization from RoBERTa’s AutoTokenizer involves breaking words into smaller units, which could fragment key political terms or hashtags. This is problematic as oftentimes political bias, especially hidden bias, relies on context clues that are only available when considering a full sentence. Therefore, we focused on preserving raw sentence structure while ensuring that our text was formatted consistently and correctly for RoBERTa embedding. Overall, our preprocessing strategy strikes a balance between cleaning the text and considering important sentence structure from social media posts. This approach allows our bias detection model to operate on authentic, minimally altered social media content, giving us the best chance to quickly and accurately return results to our users.

3.5 Bias Detection

The detection of political bias in text is a complex problem that requires a multi-faceted approach, combining advanced natural language processing (NLP)

techniques with statistical methods for uncertainty estimation. The methodology developed in this research follows a structure designed to process textual data, extract meaningful representations, and classify the data into categories. By leveraging transformer-based models, such as RoBERTa, and unsupervised learning techniques like K-Means clustering, this algorithm aims to provide a structured and interpretable framework for bias detection.

At the core of this approach lies the ability to transform raw textual input into numerical representations that capture semantic and contextual relationships. Rather than relying on predefined keywords or simplistic sentiment analysis techniques, this framework extracts deep contextual embeddings from each sentence, allowing for a more nuanced interpretation of political discourse. These embeddings are then analyzed using clustering techniques that segment data into distinct groups.

3.5.1 Embedding Generation

Once textual data has been preprocessed, the next critical step in the PolitScope pipeline is the generation of numerical representations for each sentence, a process known as embedding generation. This step is essential for converting raw text into a structured format that machine learning algorithms can analyze effectively. To accomplish this, PolitScope leverages RoBERTa (Robustly Optimized BERT Approach), a transformer-based deep learning model designed to capture nuanced semantic and contextual relationships within text [19].

Detecting political bias is inherently challenging because bias itself is a human and social construct, shaped by cultural norms, historical contexts, and individual perspectives. Traditional approaches to bias detection often rely on predefined keyword lists or sentiment scores, which introduce significant developer-imposed bias by assuming fixed definitions of politically charged language [35] [10]. To mitigate this issue, PolitScope was designed to leverage NLP techniques that minimize manual intervention and instead allow machine learning models to independently recognize patterns of bias. By utilizing RoBERTa, PolitScope learns from large-scale textual data rather than relying on human-curated rules. This methodology ensures that the detection process remains as objective and data-driven as possible, reducing the risk of coding subjective views into the system. Instead of dictating what constitutes bias, PolitScope enables unsupervised learning techniques to analyze political discourse dynamically, uncovering hidden structures in text without direct human influence.

RoBERTa operates by taking the input sentences and projecting them into high-dimensional vector spaces, where each dimension can be thought of as encoding some aspect of the sentence’s meaning. RoBERTa first takes in the textual data, which are then transformed into numerical representations through the model’s deep neural network [23]. This network, which is pre-trained on over 160GB of uncompressed diverse text, allows PolitScope to extract linguistic patterns, which we are able to apply to political discourse. One of the key benefits of using RoBERTa is its ability to understand words in context rather than in isolation. This ensures that terms with multiple meanings—such as

“right” (as in the direction) and “right” (as in the American political ideology) are interpreted correctly within their respective sentences.

To efficiently process large datasets, PolitScope implements batch processing, where multiple sentences are tokenized and embedded simultaneously. This is an important step for optimizing memory usage and computation time, enabling the system to handle large sets of sentences without excessive CPU consumption. Once the text is processed, it is passed through RoBERTa’s neural network, which produces hidden state representations for each sentence. These hidden states are then averaged to create a single sentence-level embedding vector, which serves as the final numerical representation of the given text; which is what we are interested in. The advantage of this method is that it ensures each sentence is represented as an information-rich vector that captures its full meaning, and that a computer can process.

A distinguishing feature of PolitScope’s embedding generation process is the incorporation of confidence scoring, which helps assess the reliability of the model’s representations. This is achieved by applying softmax normalization to the RoBERTa embeddings, calculating a probability distribution for each embedding vector. The softmax function, shown below, converts the vectors of raw numerical embeddings, into a probability distribution. The exponentiation of each score ensures that all values are positive, while the denominator normalizes them by the sum of all exponentiated scores in the sequence. This process ensures that each value falls within the range zero to one, and that the sum of all probabilities equals one. By doing so, softmax highlights the most dominant embeddings while still accounting for weaker signals in the data, or in other words, provides a way to measure confidence in the embedding vector. Lower entropy values indicate higher confidence, while higher entropy suggests greater uncertainty. This confidence scoring mechanism is important for later classification, as it allowed us to incorporate a level of validation to our bias detection before informing users.

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$$

Figure 3: Softmax Function

The output of this embedding process is a structured dataset where each sentence is associated with both its vector representation and confidence score. These embeddings form the foundation for future analysis, including clustering comparisons. By implementing RoBERTa’s deep contextual embeddings alongside a confidence scoring, PolitScope ensures that its bias detection algorithm is both complex yet transparent, providing users with a well-founded and interpretable analysis of political bias.

3.5.2 Model Development: Clustering Embeddings

With the embeddings generated, the next step in the PolitScope pipeline is to analyze these numeric representations and identify patterns within the data. While embeddings provide a powerful way to capture the semantic and contextual nuances of text, they do not inherently identify bias, which is our ultimate goal. To make sense of the vast amount of embedded data, we used clustering techniques that allow similar sentences to be grouped together based on their proximity, which is determined by their underlying linguistic structures. This process is essential for distinguishing patterns within political text, enabling PolitScope to recognize how different types of speech relate to one another. By applying unsupervised learning methods, we ensure that PolitScope can identify clusters of politically charged language without relying on predefined labels, maintaining a data-driven approach to bias detection.

To ensure PolitScope effectively captures patterns in political discourse, it was essential to train the model on a dataset containing both political and non-political text. If the model were trained exclusively on political speech, it would lack a reference point for identifying what distinguishes political rhetoric from general discourse. In order to avoid this we combined two free datasets from Kaggle, to get a combined dataset that includes political (around 1,727,000 tweets) and non political text (around 80,000 unique emails)[14] [20]. By including both types of text, the model can develop a more nuanced understanding of speech patterns, recognizing which linguistic features are unique to political discussions and which are more universally present. This comparative approach is crucial for avoiding misclassification, as it prevents the model from assuming that complex or opinionated language is inherently political. Instead, it learns to differentiate political bias based on structural and contextual patterns rather than superficial characteristics.

Once the embeddings are generated, the next step in the pipeline is to group similar embeddings together using K-Means clustering. This technique was chosen because it effectively identifies patterns in high-dimensional data, which is essential for analyzing complex sentence embeddings. K-Means operates by partitioning the dataset into k clusters, where each cluster represents a distinct grouping of semantically similar sentences. The algorithm begins by selecting 2 random points as cluster centroids and iteratively refines these centroids by assigning each embedding to the nearest centroid and recalculating cluster centers until convergence is reached. This process ensures that embeddings with similar vectors, and hence similar linguistic structures and contextual meanings, are grouped together, allowing for the identification of underlying trends in political and non-political speech. By structuring the data in this way, PolitScope develops a deep pattern-based understanding of political rhetoric. This is a complex procedure for a vector size of around 760 dimensions, from RoBERTa, but Figure 4 represents a simplified 3D visualization of this process. The smaller circles represent a single sentence vector from RoBERTa, simplified into the coordinates of a 3D space. The red circle represents sentences that have similar embeddings are located close together, and have been clustered together using

K-Means.

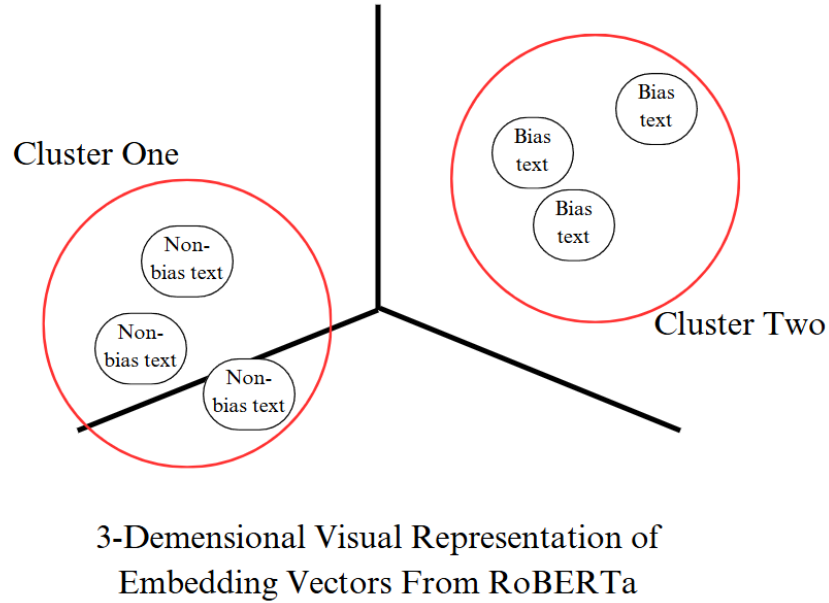


Figure 4: 3-Dimensional Visual Representation of Embedding Vectors from RoBERTa

A key advantage of using K-Means clustering is that it does not require pre-defined categories for political bias, reducing the risk of introducing developer-imposed assumptions. Instead of labeling data based on subjective criteria, K-Means allows the structure of the data itself to dictate how sentences are grouped, reinforcing the project’s goal of minimizing human-introduced bias. However, selecting an appropriate value for k is a critical consideration, as too few clusters may oversimplify the data, while too many could lead to overfitting. PolitScope uses 2 clusters as a starting point, as this binary classification distinguishes between political bias and non-political bias. While this choice offers an easily interpretable framework, it is still appropriate given the specific goals of the project; especially considering the model is designed to detect bias in a broad context, distinguishing general political discourse from non-political speech. Two clusters allow for clear differentiation between these two texts, and the simplicity of this approach helps avoid unnecessary complexity while maintaining interpretability. While the current setup of PolitScope offers a high-level overview, it comes with the drawback of potentially oversimplifying the underlying complexity of political bias. Future work on PolitScope could involve experimentation with varying values of k , balancing granularity and interpretability to better capture the nuances of political bias. The flexibility of

K-Means also allows for dynamic adjustments, making it adaptable to future refinements or expansions of the dataset.

Once the K-Means clustering process is complete, using Python’s pickle library, we save a model for future use. This model includes key components such as the trained K-Means object, which contains the cluster centers and the assignment of each sentence to a cluster. By serializing this data, we can efficiently store and retrieve the model for later analysis or comparisons without the need to re-run the entire clustering process. Saving the model in this way ensures that the embeddings, cluster assignments, and other important attributes are preserved, enabling easy access to the data for further experimentation or deployment in future iterations of the tool.

Given that PolitScope is also being integrated as a Chrome extension for ease of use by end-users, the need to save the model becomes even more critical. The primary objective for users is to quickly analyze text for political bias, without requiring them to wait for extensive processing times. Running the training and embedding steps each time a user interacts with the extension would result in significant delays, hindering the user experience. By saving the model after the initial training phase, we eliminate the need to recompute embeddings and clustering for every interaction. Instead, the pre-processed model can be quickly loaded from storage, allowing users to access results instantly and without long computational wait times.

This approach of front-loading the computational time upfront and saving the model is crucial for creating an efficient and user-friendly experience. It enables the tool to be deployed with minimal latency, as the heavy lifting is done during the initial training phase, while subsequent usage is optimized for speed. The serialized model ensures that the extension can access the data quickly, providing near-instant feedback to the user and allowing for efficient integration into everyday use. This efficiency is essential for a smooth user experience, especially in the context of a web-based application where performance and responsiveness are critical for user satisfaction.

3.5.3 Cluster Comparison

Once the PolitScope model is trained and saved, it becomes the core component for analyzing new text inputs. When a user activates the extension, the text from their active webpage is sent to the backend for analysis. This text is then processed through the initial steps of the PolitScope pipeline, including cleaning and embedding vector creation, ensuring that the input is formatted in a way that the model can understand. Putting the user provided text through these steps makes the text comparable to the clusters generated during the training phase.

At this point in the process, the embeddings of the user’s input text are ready for comparison with the previously created clusters. The goal is to identify whether any of the user-provided sentences exhibit political bias. To achieve this, the system compares the newly generated embeddings to the model’s cluster centers, determining which cluster each sentence is closest to. As shown be-

low, in order to do this, we use Euclidean distances to measure the straight-line distance between each new embedding and the centers of the existing clusters [34]. This method helps assess how similar the new sentence embeddings are to the cluster centers. As seen in Figure 5, the smaller the Euclidean distance, the closer the sentence is to that cluster, which suggests that the sentence shares similar characteristics to others within that group. This approach effectively helps categorize the new text into the most appropriate cluster based on its proximity to the cluster centers. The idea behind this approach is that the model clusters sentences based on patterns and similarities in language, and a sentence belonging to a certain cluster should exhibit certain characteristics of the broader group it is associated with.

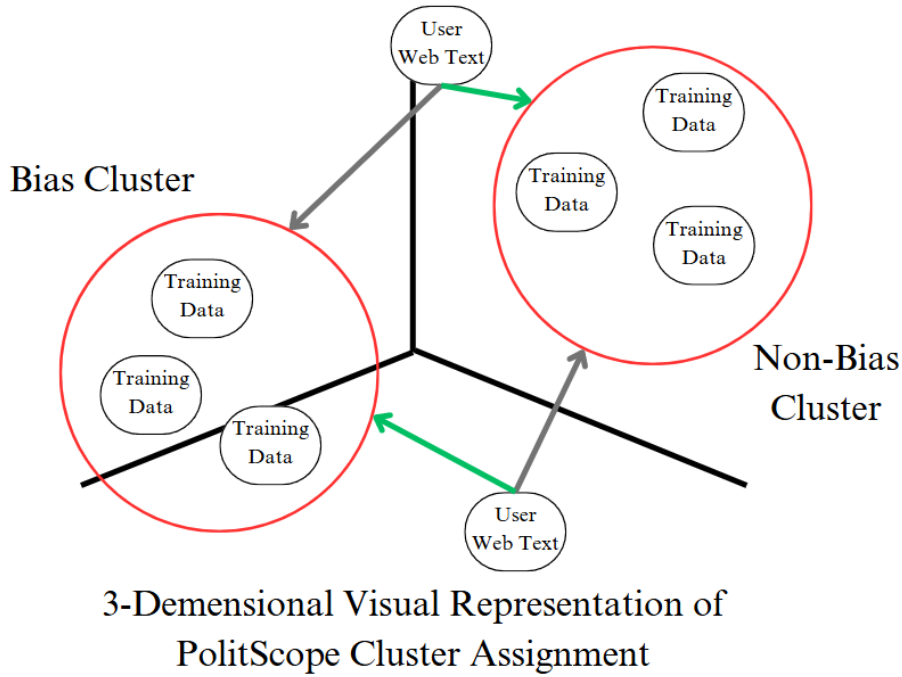


Figure 5: 3-Dimensional Visual Representation of PolitScope Cluster Assignment

While the model can automatically perform this comparison, there is an important caveat: the identification of which cluster corresponds to “bias” requires human input. In PolitScope’s current iteration, we needed to make a subjective decision regarding which cluster would represent biased content. This step introduces a controlled level of human influence into the tool but is crucial for ensuring that the model’s output aligns with our understanding of political bias. This decision was based on the nature of the training data and the features observed in the clusters; we assigned the second cluster as the “bias cluster,” while

the first cluster represented non-political or neutral content. Though this step involved human judgment, the model itself operates in a way that minimizes subjectivity and focuses on the patterns in the data.

Once the comparison of embeddings to cluster centers is complete, the system calculates a confidence score for each sentence. The confidence score is a weighted combination of two factors: the distance between the sentence’s embedding and the closest cluster center, and the entropy of the sentence’s embedding, which represents the uncertainty of the model’s prediction. The distance score is computed by measuring how close the inputted sentence’s embedding is to one of the model’s cluster centers, with a lower distance indicating a higher confidence in the assignment. Additionally, the entropy score reflects how certain the model is about the embedding’s classification, with lower entropy suggesting a more confident decision. These two factors are then combined using a weighted average, where the entropy score contributes more heavily (90%) to the final confidence calculation, and the distance score contributes the remaining 10%. This balance is explored in greater detail in the experiments chapter.

The threshold for confidence is set at 80%, meaning that only sentences with a confidence score above this value are classified as “biased.” This ensures that the system does not classify ambiguous or borderline cases as biased, reducing the risk of false positives. Once sentences are classified as biased, they are packaged up and returned to the frontend, where they can be highlighted for the user. This process allows users to quickly identify potentially biased content on their webpage and engage with it more critically, all while minimizing the computational time required for each analysis.

By combining the K-Means clustering with distance and entropy-based confidence scoring, PolitScope aims to strike a balance between classification accuracy and performance. The clustering process ensures that sentences are grouped based on patterns in the data, while the confidence scoring helps filter out uncertain cases. This methodology allows PolitScope to operate quickly and efficiently, providing users with instant feedback on whether the text they are viewing contains potentially biased content. It also opens the door for future improvements, such as experimenting with different clustering strategies or expanding the dataset to include more nuanced types of political bias.

3.6 User Interface Integration

The frontend implementation of the PolitScope tool was designed to provide a seamless user experience while integrating with the backend’s model of political bias in online content. The user interface (UI) was constructed using a combination of JavaScript and HTML, leveraging their flexibility to interact efficiently with the backend API and to present the results to the users in an accessible manner. Given the primary goal of the project to analyze and highlight political bias in text without much user input, our implementation approach needed to: extract content from the current webpage, interact with the backend for analysis, and then visually highlight and display the biased content to the user.

in a comprehensible format.

To achieve this, the JavaScript code was embedded into a browser extension to directly extract main visible text from a webpage. Instead of collecting all visible elements, PolitScope prioritizes primary content areas, filtering out irrelevant text like navigation menus, footers, or sidebars. The decision to refine text extraction ensures that the tool focuses on content most relevant to the user’s reading experience and minimizes noise from advertisements, metadata, or hidden elements. This step is crucial for ensuring that only the text displayed to the user, which has the highest likelihood of being read by the user, is included in the analysis. The visible text was then formatted into a structured JSON format, matching the format expected by the backend. This allows for smooth communication between the frontend and backend by maintaining consistency in data handling. By organizing the text into the specific format, where the key holds all the extracted text, the data could be easily passed to the backend for processing.

Once the backend processes the text and returns the results, including sentences identified as biased, the JavaScript code takes care of presenting this information in a user-friendly manner. Since the extracted text is cleaned to meet the specifications of the NLP model, an exact match between the backend’s identified biased sentences and the webpage’s content is not always possible. To address this, PolitScope maps the original sentence extracted from the webpage, to the new sentences that have been cleaned. The mapping allows us to match the sentences that were detected as bias to their original, unchanged format. This process ensures that sentences remain properly highlighted, even if minor textual differences exist between the backend’s processed version and the highlighted content.

Using JavaScript’s Document Object Model (DOM) manipulation capabilities, the tool highlights matched sentences in yellow directly on the webpage, providing immediate visual feedback to the user [4]. The choice of yellow for highlighting was intentional, as it does not carry direct associations with any American political party, ensuring that users can easily recognize the marked sentences without additional bias.

The next step in the UI process involves the creation of a popup interface. This popup is triggered by the user clicking on the browser extension’s icon. It then provides a list of the biased sentences identified by the backend analysis. The list of sentences is displayed in a clean, readable format. The inclusion of the popup interface serves multiple purposes: it consolidates the biased sentences for central access, provides context and transparency to the user regarding which parts of the text were flagged, and serves as a brief summary of overall bias present in a webpage. Additionally, a warning message is presented in the popup, explaining that the tool uses natural language processing NLP techniques, which are not flawless and may result in misidentifications. This message serves both as a disclaimer and an encouragement for users to critically analyze the results and engage in further research, promoting a responsible use of the tool, minimizing potential concerns.

While the front-end implementation of PolitScope plays a crucial role in

delivering results to the user, the primary focus of this research has been on the underlying bias analysis and detection, rather than optimizing the user interface. The UI design choices, such as the yellow highlighting of biased sentences and the popup interface listing flagged content, were made with usability and simplicity in mind, ensuring that users can quickly and effectively engage with the tool’s insights. However, it is important to acknowledge that this implementation is not necessarily the most refined or sophisticated UI design. Instead, it serves as a functional means of interacting with the core analytical engine, providing a straightforward way to access and interpret the bias detection results. Our design is also flexible, so if the backend changes, not much has to change on the frontend.

Ultimately, the strength of PolitScope lies in its backend processing, and the ability to leverage NLP techniques and sentence embeddings to identify and characterize political bias in online text. The front-end acts as a necessary bridge between this complex analysis and the user experience, allowing for practical real-world application. While future iterations may explore more advanced UI improvements, such as customizable highlighting, interactive visualizations, or user feedback mechanisms, the current implementation prioritizes simplicity and effectiveness in delivering PolitScope’s core insights. At this stage, PolitScope operates locally and is not yet available for widespread public use, but it lays the groundwork for broader deployment. The research’s main contribution is not in UI design, but in developing a methodology for detecting political bias through computational analysis.

3.7 API Communication

To ensure direct interactions between the Chrome extension and PolitScope’s model, our tool relies on FastAPI to handle communication between the frontend and backend. FastAPI acts as the middle man, processing a user’s webpage text and returning the analysis results in real time. This communication follows a request-response model, where the frontend sends a request containing the webpage content, and the backend processes the request before responding with the identified biased sentences.

When the extension is activated, the frontend gathers the visible text from the webpage and sends an HTTP POST request to the backend. This request contains the extracted text in a structured JSON format, which ensures that the data is formatted for the PolitScope pipeline. FastAPI, running using a local Python library for parsing, receives this request and validates the incoming data using Pydantic, ensuring it meets the expected format. Once validated, the text undergoes preprocessing, embedding, and cluster comparison, as previously discussed.

After processing the request, FastAPI generates a JSON response containing the sentences identified as biased. The frontend then receives this response and updates the webpage accordingly by highlighting the flagged sentences. An important aspect of this communication is Cross-Origin Resource Sharing (CORS), which enables the frontend, which is running within the browser, to

communicate with the backend, which running on a separate server. FastAPI includes middleware to handle CORS, ensuring that the extension can securely send and receive data without browser-imposed restrictions. This architecture allows PolitScope to efficiently process text and deliver results in real time, creating a smooth and interactive user experience.

3.8 Using PolitScope

PolitScope is currently a proof of concept, designed to demonstrate the potential of an automated bias detection tool that can evaluate online content in real time. While the primary focus of this research was to develop the bias detection model and a functional front-end interface, future work will explore how this prototype can be scaled into a fully deployable Chrome extension. For now, the system offers a working demonstration of the detection pipeline, from web content collection to bias classification and user display.

To run PolitScope, begin by downloading the **frontend** folder from the PolitScope Repository, and visiting the Chrome Extensions page. Enable **Developer Mode** in the top right corner, then select **Load unpacked** and choose the frontend directory. This will install the PolitScope extension directly onto your browser.

Next, open a terminal window, and navigate to the **backend** directory, and run the following command:

- `uvicorn main:app --reload`

This will start the backend API server. Once the server confirms that it's running and listening, the extension and backend will begin communicating. From this point, you can visit any website and PolitScope will automatically begin analyzing the textual content on that given page.

For development and debugging purposes, print statements have been left in the backend code. If you watch the terminal while browsing, you'll see different stages of the algorithm logging outputs, which can confirm that key components, such as text extraction, embedding, and classification, are functioning correctly. If you prefer not to monitor the terminal, you can simply wait a moment, and the classification results will appear directly in the tool's interface. In order to see the list of results, click on the PolitScope extension popup, and (if the tool detects bias on that page) you will likely see a similar output as shown in the Figure 6. As you can see, PolitScope includes the red warning to the users about the results, and the holistic report of bias sentences on the current webpage.

PolitScope

This tool uses NLP techniques to detect bias. It may make mistakes, so verify results and conduct further research.

however realizing this potential requires a rigorous and systematic evaluation process

choosing and implementing a set of relevant evaluation metrics tailored to your specific use case is another crucial step

Figure 6: Example PolitScope Output

3.9 Technical Stack and Library Usage

The development of PolitScope relied on a large variety libraries that each played a distinct role in building, training, and deploying the tool. Together, these technologies enabled both the core functionality of bias detection and the surrounding infrastructure required for a smooth user experience.

At the heart of PolitScope’s natural language processing pipeline are the `torch` and `transformers` libraries, which were used to implement and fine-tune RoBERTa for sentence embedding and classification. These tools provided the backbone necessary for analyzing political language at a nuanced level. To further support model training and analysis, we utilized `scikit-learn` for K-means clustering, allowing us to separate sentence embeddings into biased and neutral categories, while `scipy` and `numpy` were instrumental in measuring vector distances and managing the high-dimensional numerical data from RoBERTa.

In terms of data processing, `pandas` was used to ingest and manipulate datasets, while Python’s built-in `re` module helped perform regular expression based text preprocessing. The `typing` module supported code robustness through static type hinting.

To evaluate and visualize our experimental results, we used `matplotlib`, which enabled clear and interpretable graphical representations of our results. The `fuzzywuzzy` library also played a unique role in our experiments, allowing us to conduct similarity checks.

On the deployment side, we used `pickle` to serialize and save our trained model, minimizing computational load during inference. This allows PolitScope to deliver real time feedback to users without retraining the model every session.

FastAPI served as the communication layer between the backend and the user interface, providing a lightweight and efficient framework to pass predictions between components.

The backend logic was built entirely in Python, while the front-end was constructed using HTML and JavaScript, forming a simple but effective interface for real-time interaction. Though the current version of PolitScope runs locally, this tech stack lays a solid foundation for broader accessibility in future deployments.

3.10 Challenges

Throughout the development of the entire project, one of the biggest challenges was dealing with the concept of political bias, a fundamentally human construct, and attempting to translate it into something that computers can understand. Bias is complex and often context-dependent, making it particularly difficult for a machine learning model to identify consistently. The challenge wasn't just in detecting bias, but also developing a methodology that could understand political bias in a way that minimized developer input, keeping the system flexible. This required considerations, including what to train the model on so it could handle diverse content without overfitting to a particular viewpoint or source, creating an appropriate number of clusters, and more. Navigating these challenges highlight just how difficult it is to model human concepts like bias with machine learning techniques, and was a recurring challenge throughout the development of PolitScope.

Another significant challenge stemmed from the computational power required to create these 700+ dimensional vector embeddings. Given the complexity of extracting bias, as mentioned above, it was essential to train our model on a substantial amount of data. When each datapoint, ie. each sentence, is converted to a vector with over 700 numeric values, memory limitations became a pressing concern. To address this, we had to implement batching techniques, breaking the data into manageable chunks to ensure the system could efficiently process large amounts of text without running out of memory. While these adjustments improved the performance, they also added layers of complexity to the system. This challenge brought us to the current framework of PolitScope, where we front-load a large computational cost (in terms of time and memory) once, and store that as model. This ensures that the iterative nature on the user side runs in real time, providing quick analysis.

The technical hurdles did not end with data handling or model development. One of the most challenging aspects was ensuring smooth communication between the frontend and backend systems. The system needed to deliver real-time results, but we faced persistent issues with communicating data to and from a users webpage and the model. For example, the data passed between the layers of PolitScope's algorithm had to be correctly formatted, analyzed, and updated in real time, which led to many errors that had to be carefully debugged. This was especially challenging as issues were sometimes difficult to isolate.

In summary, the development of PolitScope was not without challenges,

many of which stemmed from the inherent difficulty of computationally modeling a complex, subjective concept like political bias. The combination of fine-tuning machine learning models, handling large datasets efficiently, and ensuring smooth system communication was demanding. These challenges required a great deal of iteration, debugging, and adaptation to make the system functional and accurate. Ultimately, our methods highlight how computational tasks involving human concepts are far from straightforward and require constant refinement and debugging to align with the complexities of real-world bias.

4 Experiments

To understand and assess the efficacy of PolitScope, the political bias detection algorithm developed in this project, two primary experiments were conducted. The first experiment validates the accuracy of the bias detection model by systematically adjusting hyperparameters used in classification, while the second experiment examines the influence of specific word categories—subjects, adjectives, and nouns—on the classification of bias. These experiments aim to provide insight into the inner workings of PolitScope, identifying key numeric and linguistic elements that drive bias classification and ensuring the reliability of its results.

4.1 Experimental Design

4.1.1 Confidence Validation

The first experiment was designed to evaluate PolitScope’s classification accuracy by systematically adjusting its internal parameters used in classification. As previously reported, sentences are classified as bias if they received a confidence score of 75% or higher. The confidence score consisted of two parts: distance and entropy. The balance between those two variables was systematically tested by adjusting coefficients responsible for putting more emphasis on a certain variable. The coefficients for entropy-based uncertainty and distance-to-cluster metrics, help filter out unreliable classifications. The hypothesis behind the experiment was that by tuning these two parameters, we could find an optimal ratio that produces the most accurate predictions. Specifically, we expected that some combinations of entropy and distance would better capture whether a sentence is politically biased or not.

A central challenge of this experiment was establishing a reliable “ground truth” to measure PolitScope’s performance. Since the concept of political bias is inherently subjective, using self-identified labels as the benchmark introduces variability, uncertainty, and personal bias. To help mitigate this, we used an external dataset that had already labeled text samples (around 2700 samples) into five ideological categories: “left,” “lean left,” “center,” “lean right,” and “right.” [29]. To align with PolitScope’s binary classification approach (biased vs. not biased), we simplified the dataset: all entries labeled “left” or “right” were grouped into a single class representing bias (assigned the value 1), and all “center” entries were labeled as unbiased (assigned the value 0). The “lean left” and “lean right” entries were removed from the experiment entirely, as they represented partial bias that would complicate the binary comparison.

Further preparation was required to make this dataset compatible with PolitScope’s detection process. PolitScope handles input at the sentence level, so each data row was cleaned to only provide one sentence per row. This ensured that the comparison between PolitScope’s predictions and the labeled data was made using inputs of the same structure and format. Once the dataset was prepared, PolitScope classified each sentence under various coefficient combina-

tions ranging from 90% distance / 10% entropy to 10% distance / 90% entropy, testing every 10% increment. For each configuration, we compared PolitScope’s output to the dataset labels and recorded the results.

To evaluate performance, we computed classification metrics that have previously been used in areas of binary NLP research: precision, recall, and the F1 score [18]. This experiment has a particular emphasis on the F-1 score, due to its ability to reflect the balance between the precision and recall. Precision, in this context, measures how often PolitScope is correct when it flags a sentence as biased. A high precision score means that when the tool predicts bias, it is rarely wrong—minimizing false positives. Recall, on the other hand, evaluates how well PolitScope identifies all actual biased sentences. A high recall score indicates that the tool is successfully capturing most instances of bias, minimizing false negatives. The F1 score combines both precision and recall into a single metric. This is crucial for a task like bias detection, where both types of errors can carry significant consequences. For example, misclassifying neutral content as biased (a false positive) can erode trust in the tool, while missing actual biased content (a false negative) undermines the tool’s core functionality. By using the F1 score, we ensured that the evaluation fairly accounted for both types of errors, allowing us to identify which confidence threshold configuration resulted in the most balanced and accurate classification performance.

This experiment is essential in addressing **RQ1**: How effectively can a tool identify political bias in online content using NLP techniques? By testing multiple configurations of PolitScope’s confidence thresholds against a labeled dataset, we directly assessed how well the tool can replicate human-labeled bias judgments using purely algorithmic approaches, K-Means and RoBERTa. The experiment served as a benchmark of how these components interact under different tuning conditions, ultimately revealing whether the model’s decision-making aligns with human perceptions of bias. The findings not only validate the tool’s functionality but also offer insights into the parameters that most influence its accuracy. This experiment demonstrates that PolitScope, when calibrated properly, can serve as an effective solution for detecting political bias in digital text.

However, this experiment has a few notable limitations. The most prominent is the subjectivity of bias labeling in the reference dataset. Even though we used an external dataset to mitigate personal bias, the original human judgments that classified text as “left,” “right,” or “center” are still inherently subjective, based on the creator’s understanding of political language. Additionally, our decision to discard “lean left” and “lean right” entries in favor of a strict binary classification, while necessary for clarity, may have oversimplified the nuanced nature of political language. Another limitation is sentence-level granularity—PolitScope evaluates text one sentence at a time, while some politically charged content relies on tone, context, or structure that spans across multiple sentences or paragraphs. Thus, sentence-level classification may miss broader contextual cues. Finally, the dataset used in the experiment was limited in size due to processing and time constraints, and future work should test across more diverse and expansive datasets to improve generalizability. Despite these challenges,

the experiment offers a foundational assessment of PolitScope’s classification capability and informs ongoing improvements to its model.

4.1.2 Linguistic Impact

The second experiment, was designed to to answer **RQ2**: How can the use of a tool like PolitScope help analyze and characterize political text online? This experiment is designed to analyze how different word categories—subjects, adjectives, and nouns—impact the detection of political bias within a sentence, as measured by PolitScope. Our hypothesis is that certain word categories contribute more significantly to perceived bias than others. Specifically, adjectives are expected to exert the strongest influence, as they directly modify the subject and shape the sentiment of a statement. However, the extent to which subjects and nouns contribute to bias detection remains uncertain and will be tested.

To conduct this experiment, a controlled dataset of biased sentences were constructed using a structured template format to ensure consistency and precision in analysis. Each sentence followed a simple and uniform structure: {subject} is a {adjective} {noun}. This structured approach allows for systematic manipulation of specific linguistic components. For example, a sentence such as “Trump is a bad president” follows this template and linguistically relies heavily on the context of the subject, adjective, and noun. By maintaining a fixed structure, variations of the sentence can be generated in a controlled manner, enabling an examination of how individual parts of speech affect the overall meaning and perceived bias. Figure 7 shows the entire dataset of potential subjects, adjectives, and nouns that we created our dataset with.

```

subjects = [
    "Donald Trump", "Joe Biden", "Ron Cole", "The mayor", "Anakan Skywalker",
    "Barack Obama", "Hillary Clinton", "Elon Musk", "The teacher", "The president",
    "Vladimir Putin", "Oprah Winfrey", "Tom Hanks", "LeBron James",
    "Albert Einstein", "Nelson Mandela", "The Dalai Lama", "Warren Buffet",
    "Jeff Bezos", "Bill Gates", "Steve Jobs", "Taylor Swift", "Beyoncé", "Rihanna",
    "Kim Kardashian", "Dwayne 'The Rock' Johnson", "Ellen DeGeneres",
    "Mark Zuckerberg"
]

adjectives = [
    "great", "honest", "intelligent", "charismatic", "dynamic",
    "courageous", "inspiring", "influential", "dedicated", "compassionate",
    "ambitious", "creative", "innovative", "visionary", "talented",
    "hardworking", "persistent", "brave", "focused", "humble",
    "respectful", "generous", "trustworthy", "strong", "outspoken",
    "neutral", "mature", "calm", "objective", "balanced",
    "reasonable", "methodical", "pragmatic", "reliable", "logical",
    "practical", "unbiased", "steady", "thoughtful", "sensible",
    "corrupt", "terrible", "controversial", "selfish", "greedy",
    "dishonest", "lazy", "unfocused", "unreliable", "arrogant",
    "irresponsible", "ignorant", "vindictive", "disrespectful"
]

nouns = [
    "president", "leader", "golfer", "Jedi", "role model",
    "actor", "entrepreneur", "scientist", "philanthropist", "author",
    "coach", "musician", "activist", "visionary", "comedian",
    "inventor", "teacher", "journalist", "investor", "director",
    "producer", "diplomat", "athlete", "social worker", "artist",
    "designer", "architect", "psychologist", "filmmaker"
]

```

Figure 7: Linguistic Impact Experiment Data

The primary goal of this experiment is to analyze the impact of modifying key parts of speech, specifically the subject, adjective, and noun, on sentence embeddings. A baseline sentence, “Donald Trump is a great president.” will be established, and variations will be created by altering the parts of speech, following the structured format outlined above. For instance, substituting the subject (Donald Trump → The mayor), modifying the adjective (great → corrupt), or changing the noun (president → golfer) introduces variations that allow for specific analyses. These controlled modifications provide insight into how different components of a sentence influence its semantic representation within a natural language processing (NLP) model.

To quantify these differences, the Euclidean distance between the embedding of each modified sentence and the embedding of the baseline sentence were computed. Sentence embeddings, derived from the PolitScope framework, using RoBERTa, capture the semantic meaning of text, and Euclidean distance served as the metric for measuring the degree of semantic shift resulting from each variation. A greater distance indicates a more significant change in mean-

ing, while a smaller distance suggests that the sentence remains relatively similar to the baseline. By analyzing these distances across multiple variations, the experiment determined how PolitScope interprets linguistic modifications, particularly within the context of political bias.

This experiment plays a crucial role in validating the PolitScope framework by testing its ability to detect and analyze patterns within structured statements. By systematically evaluating how RoBERTa processes different components of speech, the experiment provides insight into the model’s sensitivity to changes in subject, adjective, and noun usage. Furthermore, this approach directly addresses **RQ2**: How can the use of a tool like PolitScope help analyze and characterize political text online? By testing the impact of word choice on sentence embeddings, the study contributes to a deeper understanding of how NLP models identify contextual meaning in political discourse. The findings of this experiment can also reinforce confidence in PolitScope’s analytical capabilities but also enhance our ability to interpret and characterize key linguistic aspects in digital political environments.

One potential limitation of this experiment is the subjectivity of word replacement choices. While efforts will be made to select politically relevant terms, variations in word connotation and contextual associations may introduce bias in the experiment itself. Additionally, the structure of test sentences may oversimplify real-world political discourse, where complex phrasing and rhetorical devices influence perceived bias.

4.2 Evaluation

With the experimental designs in place, we now turn to analyzing the results from both the confidence validation and linguistic impact experiments. These findings help us understand how different model configurations and sentence structures influence PolitScope’s ability to accurately detect political bias. By examining the performance metrics and classification behaviors observed during testing, we can draw conclusions about the strengths and current limitations of the tool, as well as how well it fulfills its intended purpose of identifying bias in online content.

4.2.1 Confidence Validation Results

The results of the confidence validation experiment offer important insights into how effectively PolitScope classifies bias when different ratios of distance-to-modeled-cluster (D) and entropy (E) confidence coefficients are applied. As shown in Figure 8, F1 scores remained at 0.00 when distance was heavily weighted over entropy. However, performance began to increase once entropy was prioritized. The best performance was achieved at a 10% distance to 90% entropy ratio, yielding an F1 score of 60.52%, which falls within the average range according to standard evaluation benchmarks in machine learning [12]. This suggests that, for this particular model, entropy-based confidence plays a

significantly more valuable role in accurately classifying political bias than the distance-to-cluster metric.

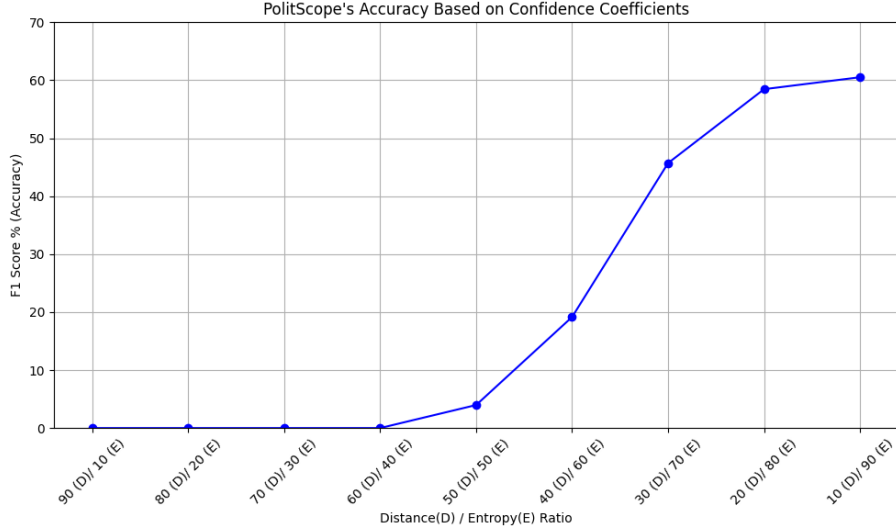


Figure 8: PolitScope’s Accuracy Based on Confidence Coefficients

These results are expected, considering the entropy score comes directly from RoBERTa’s internal probability distribution, indicating how confident the model is in its predictions. By giving more weight to this signal, PolitScope is better able to filter out low-confidence classifications and avoid false positives. On the other hand, while the distance-to-cluster metric is important to organize and differentiate biased versus neutral sentence clusters, it proved less reliable as a confidence measure during classification. This is likely due to high-dimensional overlap in RoBERTa’s sentence embeddings, where even semantically different sentences can appear close together in the embedding space. It’s important to note that in high-dimensional spaces, differences in values between sentence vectors can appear numerically small but still represent meaningful semantic distinctions. The variation between two vectors might look subtle, yet those slight shifts in embedding space often carry significant information about sentence meaning and intent. Thus, the experiment not only helps identify the best-performing coefficient configuration but also clarifies how PolitScope is arriving at its classification.

We defined a successful classification as one with an F1 score of at least 0.5, a threshold that reflects a baseline of balanced precision and recall. The 10% distance to 90% entropy configuration exceeded this threshold and is therefore considered a qualified successful configuration. However, since the F1 score did not reach the “good” range, outlined as above a 0.8, this also highlights an opportunity for future improvements [12].

4.2.2 Linguistic Impact Results

The results of the linguistic impact experiment include evidence indicating that different parts of speech contribute to the semantic representation of politically charged language, as understood by the PolitScope algorithm. By systematically modifying individual components within a controlled sentence structure and measuring the Euclidean distance between each variation and our baseline sentence, we were able to isolate and quantify the impact of subjects, adjectives, and nouns on sentence embeddings. The results, represented in Figure 9, highlight the average distance to the baseline sentence, depending on what part of the sentence was held constant: subject, adjective, and noun.

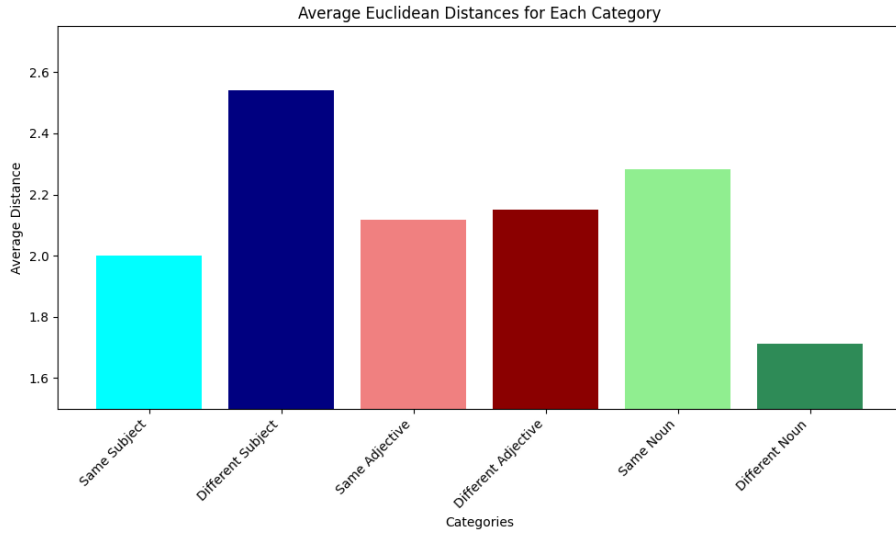


Figure 9: Subject, Adjective, and Nouns Impact on Sentence Embeddings

The results clearly indicate that the subject of a sentence has the greatest influence, with an average embedding shift of approximately 2.5 when changed, compared to just 2.0 when it remained the same. In Figure 9, the light blue bar represents 56 instances where the subject was held constant, and the adjective and noun were changed. Then the dark blue bar shows that when subject was changed and the other categories were held constant, it showed to have the largest impact of RoBERTa’s understanding of a sentence (EX: when Trump was changed to Biden, the distance changed substantially) This jump suggests that who or what the sentence is about heavily shapes how the model interprets meaning. This is likely because political figures or entities can carry strong contextual and ideological associations that directly affect perceived bias.

Adjectives, which were hypothesized to have the strongest effect due to their sentimental nature, produced a more modest difference: 2.2 when changed versus (dark red bar) 2.1 when constant (light red bar). This small change is still

meaningful, especially considering that adjectives contribute to the emotional tone of a sentence. However, the averaged results may have been dampened by the inclusion of both positive and negative adjectives, whose opposing effects could have canceled each other out numerically. What we mean by this, is that we could have conducted a similar experiment, but just looking at the effect of positive and negative charged adjectives. Since this experiment combines them into one category, their effects on PolitScope’s identification may balance each other out. Nevertheless, this finding still confirms that adjectives impact sentence meaning, just not as drastically or predictably as subjects.

The most surprising finding comes from the noun analysis. When the noun was kept constant, the average distance from the baseline was 2.3 (light green bar), but when the noun was changed, the average distance decreased to 1.75 (dark green bar). This suggests that changing the noun, but keeping subject and adjective constant, reduces the semantic divergence from the baseline. This is likely because nouns may not significantly reshape the overall ideological framing, especially if the subject and adjective remain constant. In other words, nouns appear to play a minimal role in the model’s understanding of political bias, at least within the syntactical constraints of the experiment. These experimental results show how PolitScope’s embeddings, created by RoBERTa, primarily derives semantic and ideological meaning from who is being discussed and how they are being described, rather than from the label being assigned.

This experiment successfully meets its objective by identifying that subjects disproportionately impact sentence embeddings, thereby validating PolitScope’s design to analyze structured political language. These results are critical for answering Research Question 2: How can the use of a tool like PolitScope help analyze and characterize political text online? The findings show that PolitScope can go beyond surface-level keyword analysis to detect how linguistic components interact to produce meaning and bias. Understanding which parts of a sentence carry the most ideological weight allows users and researchers to better interpret PolitScope’s predictions and even audit politically sensitive language in a more structured, data-driven way.

4.3 Threats to Validity

While the experiments conducted for PolitScope produced meaningful insights, several threats to validity, both inherent to the original research design and introduced through our implementation, must be considered. These limitations are crucial in contextualizing the findings and shaping future work on the tool.

First, the design of the confidence validation experiment, used a pre-labeled dataset. While using a pre-labeled dataset provided a structured and objective way to evaluate PolitScope’s performance, it is important to acknowledge that this approach is far from ideal. Political bias is inherently subjective, and any dataset that definitively labels content inevitably reflects the biases and assumptions of those labeling it. Moreover, using this dataset treats bias as a static attribute rather than a dynamic quality influenced by framing, audience, and interpretation. However, given the complexity and impracticality of man-

ually labeling large amounts of data in a consistent and unbiased way, relying on a pre-labeled dataset was the most feasible option. It allowed us to define clear, measurable marks for success and perform comparisons across different experimental conditions. Still, future iterations of PolitScope could benefit from different validation approaches, such as multiple judgments from human participants, to better account for the fluid and contested nature of political bias.

On top of that, the data relies on a biased to unbiased categorization. This binary-like structure excludes the nuanced ideological spectrum found in real-world discourse. For instance, data that is not categorized as either biased or not, are omitted, which flattens the diversity of political viewpoints present in the results and limits the model’s ability to detect subtle or compound biases. As a result, PolitScope may miss more nuanced expressions of ideological alignment, particularly those that are intentionally ambiguous.

Another threat arises from our use of single-sentence inputs. Political bias often depends on contextual cues that span multiple sentences, including tone shifts, rhetorical patterns, or references to prior discourse. By reducing input to single, isolated sentences, we risk underestimating the complexity of language used in political communication. This limitation inevitably limits its capacity to capture deeper or more subtle bias signals that occur over longer discourse.

Moving to the linguistic impact experiment, additional validity concerns surface. The highly controlled template structure: “{subject} is a {adjective} {noun}”—was intentionally designed to isolate and test the influence of different parts of speech. While this structure enabled clear, quantifiable comparisons across sentence types, it simultaneously oversimplifies language. Political speech is rarely so rigid or uniform. Real-world sentences often include metaphor, irony, idioms, and rhetorical devices that drastically alter interpretation. By stripping sentences down to their basic grammatical components, we may have excluded features that play a significant role in how NLPs interpret bias.

These limitations highlight the need to analyze PolitScope’s results with caution. While both experiments were carefully designed and revealed meaningful trends, they operate within controlled linguistic and ideological frames. Yet, these controlled environments were crucial in validating the tool’s internal logic, revealing areas for future refinement, and answering our research questions. Overall, although PolitScope demonstrates promising capabilities in detecting political bias through NLP techniques, the results should be viewed as an initial validation within a specific scope rather than a complete proof of correctness.

4.4 Future Research Experiments

While our two primary experiments provide valuable insights into word influence and model accuracy, future research could explore additional factors affecting PolitScope’s performance and accuracy. One area of investigation, that could directly influence correctness, is the impact of the number of clusters in K-Means classification. The current model operates with a fixed number of ideological clusters, but testing various values of k may reveal whether more or less

classifications improve accuracy.

Another direction is to test PolitScope using alternative NLP models such as DistilBERT or GPT-based embeddings. Different models may capture linguistic nuances differently, potentially affecting classification results. Comparing performance across models will determine whether RoBERTa remains the optimal choice for bias detection.

In addition to enhancing accuracy, future experiments could focus on improving PolitScope’s efficiency through experiments that measure runtime performance and memory usage. By benchmarking key components, such as embedding generation, clustering, and distance calculations, we could identify computationally expensive parts of the algorithm and test alternative implementations or optimizations. Refactoring the current approach, experimenting with parallel processing, or simplifying feature extraction could lead to faster analysis times, making the tool more scalable for real time analysis.

5 Conclusion

Now, it’s essential to step back and reflect on what this project reveals more broadly about the intersection of natural language processing (NLP) and political discourse. This study set out not only to build a tool, but to explore whether modern NLP techniques can be effectively leveraged to detect and interpret political bias in online content. While the experiments were designed to test PolitScope’s internal mechanisms, the larger takeaway lies in what these results suggest about the potential of automated approaches to political text analysis.

5.1 Summary of Results

PolitScope is a qualified success in addressing the two research questions posed at the start of this study. In response to **RQ1**: “How effectively can a tool identify political bias in online content using NLP techniques?”, the tool demonstrated a moderate level of accuracy in distinguishing bias across a variety of inputs. Though the accuracy level peaked at around 0.605, indicating that PolitScope is far from perfect, it shows that with the right tuning and future implementation, NLP models can meaningfully contribute to the identification of political bias.

As for **RQ2**: “How can the use of a tool like PolitScope help analyze and characterize political text online?”, our results confirm that NLP-based embedding analysis can offer nuanced insights into how political bias is linguistically structured. Through controlled experiments on parts of speech, we found that elements such as the subject of a sentence had disproportionate impact on how the model interprets meaning. This lays a critical foundation for building our understanding of how political framing operates. These findings validate PolitScope not just as a classifier, but as a linguistic lens through which political rhetoric can be examined in a computational way.

Ultimately, PolitScope should not be taken as an authoritative detector of bias, nor is it a perfect product. However, it does serve as a proof of concept. It demonstrates the viability of using NLP techniques to analyze political content online with a level of interpretability and structure. From initial research questions to method development, experimentation, this project illustrates how computational tools can help bring light to one of the most hidden yet consequential issues in online discourse today: political bias.

5.2 Future Work

While PolitScope represents a strong proof of concept for using NLP techniques to detect political bias in online content, there remains a wide range of improvements and extensions that could significantly increase its accuracy, utility, and overall impact. These future directions reflect both technical limitations encountered during development and broader ambitions to make PolitScope a widely accessible, real-time tool for political awareness.

One of the most immediate areas for improvement is training the model on a larger and more diverse dataset. The initial training data, though sufficient for this version, is ultimately limited in scope. Expanding the dataset to include a broader spectrum of political perspectives (including international sources, nuanced or subtle leanings and a mixture of formal articles, opinion pieces, and social media posts) would allow PolitScope to better capture the full complexity of political discourse. A more comprehensive dataset would reduce overfitting and improve generalizability, especially for ambiguous or subtle examples of political bias.

Equally important is the need for PolitScope to move beyond sentence-level classification. While the current design analyzes political text one sentence at a time for real-time performance, this approach sacrifices important contextual relationships between sentences—relationships that often influence how bias is perceived. Future iterations should include paragraph, or entire webpage modeling, enabling the tool to assess how earlier sentences frame later ones, or how language builds over time to create rhetorical bias. In online content, especially news articles or political sites, bias is rarely encoded in a single sentence. Capturing this evolving structure is key to more robust analysis.

Another potential direction is the exploration of alternative NLP models. While RoBERTa proved effective for embedding and classification tasks, it is just one of many powerful language models available today. Tools such as DeBERTa, T5, DistilBERT, or other models may yield better performance, especially in identifying subtle undertones like sarcasm, euphemism, or ideological framing. Running PolitScope’s pipeline with new NLP models could reveal new insights into which models are most sensitive to political semantics and why.

We also see substantial potential in experimenting with different clustering configurations. PolitScope currently uses two clusters to distinguish between biased and neutral content. While this binary classification is intuitive and easy to interpret, it may oversimplify political discourse. Future experiments could test whether adding more clusters could help the model surface more nuanced ideological signals. In addition to varying the number of clusters, exploring alternative clustering algorithms beyond k-means, such as hierarchical clustering or DBSCAN, could better capture the complex structure of political language. This would move the tool beyond binary judgments and toward political mapping, helping users better understand where content falls on a broader ideological spectrum.

Beyond model improvements, PolitScope’s user interface and system architecture also present opportunities for meaningful advancement. Currently, the tool runs locally and processes static webpages. A future version could be hosted on a central server, enabling broader access and faster performance across devices. Ideally, PolitScope would be distributed as a browser extension that users can install directly into Chrome or Firefox. This would allow real-time, on-page classification of political content as users browse the web. Adding functionality that allows user to create accounts could also let users track the types of bias they encounter most frequently, giving them a holistic view of their media consumption and helping them make more informed choices about the content

they consume. Additionally, improving the tool’s ability to extract only the relevant text from complex webpages, ignoring ads, footers, comments, and other irrelevant content, would make analysis cleaner and more reliable.

All of these avenues of future research emerged during the research and development process, and they highlight just how rich and open-ended the future of this project truly is. PolitScope, as it stands, is a meaningful step toward computationally understanding political bias, but it is just the beginning. With further development, PolitScope has the potential to become a powerful tool for personal media literacy, academic research, and civic engagement in a digital world filled with political content.

5.3 Future Ethical Implications and Recommendations

While PolitScope offers a promising step forward in the automated detection of political bias, its use, especially in public contexts, raises several ethical considerations. It is critical to reflect on what responsibilities come with releasing or scaling this tool, and how its results might be interpreted or misused in the real world.

One of the most significant ethical issues stems from the interpretation of our bias labels as objective truths. Despite PolitScope’s ability to classify text based on learned patterns, it is far from perfect and will likely make mistakes. The use of sentence embeddings and clustering allows for meaningful comparisons, but it also introduces the risk of oversimplification. Labeling a sentence as biased without full context may unintentionally give people the false impression that the classification is universally agreed upon or definitive. As we noted in our experimental results, the tool is sensitive to relatively small linguistic changes, which means that real-world political content may not always be captured accurately or fairly.

The public release of PolitScope, especially as a browser extension or large-scale monitoring tool, also opens the door to ethical misuse. If deployed irresponsibly, PolitScope could be used to discredit or flag legitimate claims, discredit certain authors, or add to echo chambers by validating users’ preconceived notions of what “bias” looks like. To responsibly move forward, we recommend several ethical guidelines and design constraints. First, any future version of PolitScope should include clear disclaimers about the limitations of its classifications, and ideally offer explanations to help users understand why a certain label was applied. Secondly, a browser extension of PolitScope should be opt-in, transparent, and respectful of user privacy, with no personal data stored or shared without explicit consent. Lastly, collaborating with political scientists, ethicists, or media scholars would ensure the tool remains grounded in responsible, ethical practice and doesn’t inadvertently contribute to the very biases it seeks to expose.

In conclusion, PolitScope demonstrates that NLP can be used to surface and analyze patterns of political bias, but its power must be matched by a deep understanding of ethical concerns. If PolitScope moves forward into more advanced models, broader datasets, and user deployments, it is crucial to include

ethical reflections into every layer of development.

References

- [1] 2021. Unsupervised Learning: Algorithms and Examples. *altexsoft* (2021). <https://www.altexsoft.com/blog/unsupervised-machine-learning/>
- [2] 2022. Political digital advertising spending in the United States between January 1 and August 1, 2022, by type. *Statista* (2022). <https://www.statista.com/statistics/1325158/political-digital-ad-spending-usa/>
- [3] 2022. Political Polarization: A Clear and Present Danger. *Journal of Sociology and Christianity* 12, 1 (2022), 5.
- [4] 2023. Document Object Model (DOM). mdn. https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model
- [5] 2024. Average time spent on selected social media platforms daily among adults in the United States in February 2024, by age group. *Statista* (2024). <https://www.statista.com/statistics/1484565/time-spent-social-media-us-by-age/>
- [6] 2024. Toxic Polarization: The Latest Numbers. *L1STEN First Project* (2024). <https://www.listenfirstproject.org/toxic-polarization-data>
- [7] 2025. How fact-checking works. *Meta* (2025). <https://transparency.meta.com/features/how-fact-checking-works>
- [8] 2025. Misinformation, Disinformation, and Bias. *Eastern Michigan University Library* (2025). <https://guides.emich.edu/misinformation/readerbias>
- [9] Mohiuddin Ahmed, Raihan Seraj, and Syed Mohammed Shamsul Islam. 2020. The k-means Algorithm: A Comprehensive Survey and Performance Evaluation. *Electronics* 9, 8 (2020). <https://www.mdpi.com/2079-9292/9/8/1295>
- [10] Mohd Zeeshan Ansari, M.B. Aziz, M.O. Siddiqui, H. Mehra, and K.P. Singh. 2020. Analysis of Political Sentiment Orientations on Twitter. *Procedia Computer Science* 167 (2020), 1821–1828. <https://doi.org/10.1016/j.procs.2020.03.201> International Conference on Computational Intelligence and Data Science.
- [11] David P. Baron. 2006. Persistent media bias. *Journal of Public Economics* 90, 1 (2006), 1–36. <https://doi.org/10.1016/j.jpubeco.2004.10.006>
- [12] Nikolaj Buhl. 2023. F1 Score in Machine Learning. *Encord* (2023). <https://encord.com/blog/f1-score-in-machine-learning/#:~:text=Typically%2C%20an%20F1%20score%20%3E%200.9,to%20have%20a%20poor%20performance>

- [13] Risa Gelles-Watnick Colleen McClain, Monica Anderson. 2024. How Americans Navigate Politics on TikTok, X, Facebook and Instagram. *Pew Research* (2024). <https://www.pewresearch.org/internet/2024/06/12/how-americans-navigate-politics-on-tiktok-x-facebook-and-instagram/>
- [14] Noah Finberg. 2021. Political Emails. *Kaggle* (2021). <https://www.kaggle.com/datasets/noahfinberg/political-emails>
- [15] Robin Geuens. 2023. What is the average time spent on social media each day? *Soax* (2023). <https://soax.com/research/time-spent-on-social-media>
- [16] H2O.ai. 2024. What is BERT and how is it Used in AI? <https://h2o.ai/wiki/bert/>
- [17] Facing History and Ourselves. 2024. Political Polarization in the United States. (2024). <https://www.facinghistory.org/resource-library/political-polarization-united-states>
- [18] Jane Huang. 2024. Evaluating Large Language Model (LLM) systems: Metrics, challenges, and best practices. *Medium* (2024). <https://medium.com/data-science-at-microsoft/evaluating-llm-systems-metrics-challenges-and-best-practices-664ac25be7e5>
- [19] Hugging Face. 2024. RoBERTa Model Documentation. https://huggingface.co/docs/transformers/en/model_doc/roberta Accessed: 2025-05-05.
- [20] Manch Hui. 2021. US Election 2020 Tweets. *Kaggle* (2021). <https://www.kaggle.com/datasets/manchunhui/us-election-2020-tweets>
- [21] IBM Editorial Team. n.d.. Unsupervised Learning. <https://www.ibm.com/think/topics/unsupervised-learning>.
- [22] Kalaiarasi Sonai Muthu Anbananthen Kian Ming Lim Kian Long Tan, Chin Poo Lee. 2022. RoBERTa-LSTM: A Hybrid Model for Sentiment Analysis With Transformer and Recurrent Neural Network. *IEEE Access* 10 (2022). <https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9716923>
- [23] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Ro{BERT}a: A Robustly Optimized {BERT} Pretraining Approach. <https://openreview.net/forum?id=SyxS0T4tvS>
- [24] Jennifer McCoy. 2019. Polarization harms democracy and society. *International Catalan Institute For Peace* (2019). <https://www.icip.cat/perlapau/en/article/polarization-harms-democracy-and-society/>

- [25] Genie Barton Nicol Turner Lee, Paul Resnick. 2019. Algorithmic bias detection and mitigation: Best practices and policies to reduce consumer harms. *Brookings* (2019). <https://www.brookings.edu/articles/algorithmic-bias-detection-and-mitigation-best-practices-and-policies-to-reduce-consumers-harms/>
- [26] Warje K. Pilat D., Sekoul D. 2021. Negativity Bias. *The Decision Lab* (2021). <https://thedecisionlab.com/biases/negativity-bias>
- [27] Prabhu Raghav. 2019. Understanding NLP Word Embeddings — Text Vectorization. *Medium* (2019). <https://medium.com/data-science/understanding-nlp-word-embeddings-text-vectorization-1a23744f7223>
- [28] Dan Robitzski. 2019. Ex-Googler: Company Has "Voodoo Doll, Avatar-Like Version of You". *The Byte* (2019). <https://futurism.com/the-byte/google-company-voodoo-doll-avatar>
- [29] Mayobanex Santana. 2025. Political Bias. *Kaggle* (2025). <https://www.kaggle.com/datasets/mayobanexsantana/political-bias>
- [30] Shayan Sardarizadeh. 2019. Instagram fact-check: Can a new flagging tool stop fake news? *BBC News* (2019). <https://www.bbc.com/news/blogs-trending-49449005>
- [31] Pew Research Staff. 2014. Political Polarization in the American Public. *Pew Research* (2014). <https://www.pewresearch.org/politics/2014/06/12/political-polarization-in-the-american-public/>
- [32] Evan Sunwall Tim Neisser. 2023. Error-Message Guidelines. *Nielsen Norman Group* (2023). <https://www.nngroup.com/articles/error-message-guidelines/>
- [33] Brown University. 2020. U.S. is Polarizing Faster Than Other Democracies; Study Finds. *News From Brown* (2020). <https://www.brown.edu/news/2020-01-21/polarization>
- [34] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17 (2020), 261–272. <https://doi.org/10.1038/s41592-019-0686-2>
- [35] Hao Wang, Dogan Can, Abe Kazemzadeh, François Bar, and Shrikanth Narayanan. 2012. A System for Real-time Twitter Sentiment Analysis of

2012 U.S. Presidential Election Cycle. In *Proceedings of the ACL 2012 System Demonstrations*, Min Zhang (Ed.). Association for Computational Linguistics, Jeju Island, Korea, 115–120. <https://aclanthology.org/P12-3020/>