SvedbergOpen
DISSEMINATION OF KNOWLEDGE
International Journal of Artificial Intelligence and Machine Learning
Publisher's Home Page: https://www.svedbergopen.com/
Research Paper Open Access

Enhancing Social Media Bias Detection With Contextual N-Grams

Mr. Prasanth G Rao[1*], Harsha Chigurupati[2], Krish Hashia[3], Dr. Thriveni J[4], Dr. P Deepa Shenoy[5], Dr. Venugopal K R[6]

[1*]Research Scholar, Department of Computer Science and Engineering, University Visvesvaraya College of Engineering, India, 560001, email id: prasanthgrao@gmail.com, ORCID id: 0009-0008-9507-2264
[2]Undergraduate Student, Department of Computer Science and Engineering, University Visvesvaraya College of Engineering, India, 560001, email id: chigurupatiharsha2004@gmail.com, ORCID id: 0009-0001-0575-4157
[3]Undergraduate Student, Department of Computer Science and Engineering, University Visvesvaraya College of Engineering, India,560001, email id: krishhashia19@gmail.com, ORCID id: 0009-0000-1855-0965
[4]Professor, Department of Computer Science and Engineering, University Visvesvaraya College of Engineering, India, 560001, email id: thrivenijgowda@yahoo.co.in, ORCID id: 0000-0002-4307-6318
[5]Professor, Department of Computer Science and Engineering, University Visvesvaraya College of Engineering, India, 560001, email id: shenoypd1@gmail.com, ORCID id: 0000-0003-0113-1419
[6]Professor, Department of Computer Science and Engineering, University Visvesvaraya College of Engineering, India, 560001, email id: venugopalkr@gmail.com, ORCID id: 0000-0002-4513-8949

ABSTRACT
Detecting bias in social media text requires methods that remain reliable under severe class imbalance while preserving interpretability. This study evaluates contextual n-gram bias detection using a unified probabilistic log-likelihood framework with Laplace smoothing. Five variants are compared across four benchmark datasets: Z-Score Baseline, NO_Trigram (unigram baseline), TRIG_NOPOS, TRIG (POS-aware), and PENTA. Results show that simpler lexical representations, especially unigram features, provide the most stable minority-class detection, whereas higher-order contextual encodings increase sparsity and reduce precision-recall balance in skewed settings. Although richer n-gram windows add expressiveness, they amplify estimation noise and do not consistently improve discrimination. On moderately imbalanced datasets, NO_Trigram achieves the strongest overall balance; on highly skewed data, all fixed-window n-gram models show degradation, with complex variants failing to deliver consistent gains over the baseline. These findings demonstrate that increasing n-gram complexity alone is insufficient for robust bias detection and motivate alternative context representations that can capture nuance without sacrificing statistical reliability.

Keywords: Bias detection; Social media; N-gram; Log-likelihood; Class imbalance

1. INTRODUCTION

Digital conversations and algorithms are influential in our perception of reality and knowledge and can be dangerous if they are biased on social media platforms [1,2]. These platforms are not just a mirror of society, but platforms which help to make it real [3]. This is because the identification, measurement and re-reduction of hidden biases is crucial in both technical and understanding of knowledge creation, acceptance and sharing [4–6].

Recent research has started to examine the ideas and beliefs behind algorithmic systems. Richter et al. [7] show how different national stories about artificial intelligence in the United States, China, and Germany lead to different views of AI's role in society and how it shapes knowledge. These stories make it clear that AI technologies are not neutral tools; they are shaped by culture and ideology. Similarly, Tsimpoukis [8] looks at how dominant and alternative views on AI have developed in the French press and on social media. He highlights how tech companies and governments help shape public opinion. His findings show how bias on social media can create strong but sometimes misleading pictures of technology, increasing political division and confusion about what is true.

In a previous study [9] we presented a Bayesian approach for detecting bias, based on the differences in word use between different datasets as evidence for ideological differences. Although this method was simple and easy to explain, it did not capture more complex language aspects such as grammar, common phrases, and word relationships that are often important for identifying subtle and/or hidden bias. This is overcome in this work by using a probabilistic log-likelihood framework with Laplace smoothing for

principled estimation of feature class association, regardless of whether the representation is dense or sparse, in the form of n-grams. With this change, one can effectively and equitably test contextual n-gram models for varying degrees of class imbalance.

In addition, we use a collection of n-gram models (trigram, trigram without part-of-speech tagging and pentagram) which are intended to capture grammar and meaning on word sequences. Different variants employ different representational strategies, enabling us to investigate the impact of varying levels of granularity of the context on bias detection, feature sparsity, and downstream classification performance in realistic and class-imbalanced conditions.

In this paper, we refer to the computational implementation referred to as the NO_Trigram model as the Unigram Baseline. This model is the baseline lexical model which is the usual unigram representation and is used as a comparison for the contribution of higher order contextual features. This paper is organized as follows: In the Introduction, background, related work and the problem statement. The method and implementation details are described in Section 2. Results and discussion are presented in Section 3. Last, Section 4. brings the paper to a close and provides future directions.

1.1.Literature Survey

Ahmed et al. [10] performed a thorough exploration of different algorithms (e.g., Support Vector Machine (SVM), Stochastic Gradient Descent (SGD) with various features (e.g., Term Frequency-Inverse Document Frequency (TF-IDF), n-gram). The results of the evaluation, which was done with 5fold cross validation, showed that the TF-IDF method significantly improved the classification results, and that the lower order n-grams (such as unigrams and bigrams) gave the best results.

To tackle the problem of aspect category detection in multilingual online reviews, Ghadery et al. [11] proposed a novel architecture called Multilingual N-gram based Convolutional Network (MNCN). The model combines n-gram analysis, convolutional neural networks, and multi-lingual word embeddings to enable processing of reviews simultaneously in multiple languages. MNCN was evaluated on the SemEval multilingual dataset, achieving high performance on the multilingual data, which is a pioneering approach that performs aspect category detection simultaneously in multiple languages.

The Lexicon-pointed Hybrid N-gram Feature Extraction Model (LeNFEM) proposed by Ogunwale et al. [12] is another new model that integrates lexical, syntactic, and semantic features for sentence-level sentiment analysis. The model is based on a fixed size window of three words (trigram) around sentiment-relevant words detected using a lexicon, and thus represents a hybrid feature set that combines word tokens, part-of-speech (POS) tags and semantic orientations. The combination of n-gram patterns and POS tagging is found to be a good contextual representation which is more powerful than the conventional Bag-of-Words model and pure n-gram model. Further, the authors reduce the feature space using another technique called Minimum Redundancy Maximum Relevance (MRMR). Experimental results for benchmark data sets (Yelp, IMDb and Amazon) showed that LeNFEM (particularly when combined with an SVM classifier) and binary occurrence vectorisation gave better F-measure scores. The results were statistically validated using the Wilcoxon Signed-Rank Test, which showed that the model was effective and the hybrid feature engineering approach was significant in sentiment analysis tasks.

Ridwan et al. 2022 [13] performed sentiment analysis of tweets related to floods using Naïve Bayes classifier and n-gram features. The research was conducted with the aim of categorizing public opinions expressed on Twitter into relevant and irrelevant sentiments about the disaster in the year of 2021 when the flood occurred in South Kalimantan. The study used n-gram features (n=1, n=2, n=3) to evaluate the performance of different text representations to improve the classification accuracy. The results showed that bigram features gave better accuracy than unigrams and trigrams, which showed that it is important to capture the bigram features.

Word pairs for sentiment analysis in context. The results showed that the method has the potential to classify the disaster related sentiments in social media by using the probabilistic classifiers along with the n-gram features.

VVu and Le [14] proposed a framework for identifying hate and offensive spans in Vietnamese social media text named ViHateOff, which uses an automatically built Hated Speech Dictionary (HSD) and the PhoBERT-large language model. They first extract a dictionary of offensive syllables and words from annotated data, then use a masked attention mechanism, which doubles the weights of those tokens matched by the dictionary, during the classification process, so that the model is more sensitive to known indicators of dangerous content. Experimental results showed an F1-score of 0.8693 for all-spans subset and an F1-score of 0.8709 for multiple-spans subset, which are relative improvements over the best baseline of more than 10%. The dictionary-based feature augmentation strategy used in ViHateOff is conceptually similar to our n-gram features filtering with a WordNet-derived hash table, and shows that lexical resources continue to be useful in directing detection models to linguistically meaningful representations.

To detect offensive behavior and cyberbullying on Twitter, Yafooz et al. [15] proposed a scalable approach that utilizes textual, user-related, and network-related features in a multichannel deep learning

framework using BiGRU, transformer blocks, and CNN. They tested their model on three commonly used datasets of hate speech, and they managed to reach around 98.95% accuracy in the classification of the comments as offensive or non-offensive. In particular, the authors used text representation methods such as bag-of-words, TF-IDF, and word embeddings, showing that neural architectures are superior at learning context semantics, but that the type of feature representation is crucial for classification performance when dealing with class imbalance, as we have found, richer n-grams do not necessarily lead to better discrimination under class imbalance.

Harish et al. [16] introduced a semantic-syntactic graph framework for aspect-based sentiment analysis, called SentSemSynNet, that combines contextual semantic representations and syntactic dependency modeling with a single graph-neural formulation. They adopt BERT for contextual encoding, BiL-STM for sequence dynamics, and a two-layer GCN for dependency propagation to classify the sentiment in sentences containing multiple conflicting sentiment cues for each aspect. The model achieved 88.25% accuracy and 82.95 macro-F-score on the restaurant task and 84.52% accuracy and 80.26 macro-F-score on the laptop task on SemEval-2014, suggesting that the model was better able to discriminate at the fine-grained level when semantic and syntactic information are modeled together.

Ziaulla and Biradar [17] introduced a hybrid LLM-assisted ABSA architecture (H-BERT) which combines SpanBERT, BiLSTM and CRF, and further leverages LLM-generated synthetic data to enhance coverage of aspect patterns beyond the scope of traditional benchmark annotations. They achieved an accuracy of 90.58% and a F-score of 90.56% on the laptop domain and an accuracy of 91.21% and a F-score of 92.03% on the restaurant domain in the SemEval-2014 domains, indicating that the hybrid contextual modeling with sequence labeling and augmented supervision can be effective for the aspect extraction and polarity classification in fine-grained sentiment tasks.

Emilio Ferrara [18] gave an in-depth overview of the many facets of bias in AI systems, focusing on the new and growing concerns surrounding generative AI models. The research systematically uncovered the key biases in the data, algorithmic biases, human decision-making biases and the societal implications of these, such as the ability to reinforcement of inequalities and the dissemination of negative stereotypes. Ferrara discussed many of the mitigation strategies that were employed, as well as data preprocessing, model selection, and post-processing, and mentioned the ethical issues and trade-offs that occurred with these mitigation strategies. The survey revealed that interdisciplinary collaboration is crucial and the need for more comprehensive strategies to ensure fairness and accountability of the use of AI systems, especially in the context of generative AI use in content generation and decision-making.

Hovy et al. [19] gave a detailed overview of the various sources of bias in NLP systems. They found five components: Data selection, Processes of annotation, Input representations, Model architectures, and Research design choices. They observed that n-gram models in general tend to reflect and perpetuate social biases as they are based on “co-occurrence” at the surface level and not contextually. The authors showed that n-gram models trained on the basis of unbalanced corpora are prone to the preservation of common stereotypes and that these models can be conditioned to learn the original stereotype. The authors demonstrated that a set of models based on n-gram models that are trained on unbalanced corpora can reinforce such stereotype, and that the original stereotype can be reproduced by these models.

Literature review reveals that the n-gram models have been successful in other related tasks like fake news detection [10], multilingual aspect classification [11] and sentiment analysis [12, 13] and fairness-aware NLP [18, 19]. These works, however, do not directly address the relationship between n-gram complexity with class imbalance, a specific problem in the context of bias detection in social media. Studies which criticize on the surface level [19] do it theoretically instead of by means of controlled empirical comparison of encoding strategies. A more practical question that remains unanswered is whether the introduction of contextual and syntactic granularity to n-gram representations leads to an increase in bias detection under realistic, skewed data distributions, or if the impact of the increased sparsity of the features is outweighed by the gains in expressiveness? This work tackles that issue head on.

It is worth noting that several of the studies discussed above [14–17] achieve strong classification performance precisely by incorporating Transformer-based components such as BERT, PhoBERT, or SpanBERT. We deliberately do not adopt such architectures in this work. Our primary concern is explainability at scale: a bias-detection system intended for continuous deployment over social media streams must be able to justify, for any single flagged post among millions, exactly which words or phrases drove the decision, in a form a human moderator or auditor can inspect without additional tooling. Transformer-based classifiers distribute a prediction across millions of contextualized, entangled parameters, so recovering that kind of per-decision, human-verifiable justification is fundamentally harder than with a feature-based model, regardless of how much compute is available to train or serve them. Our log-likelihood framework, by contrast, assigns each feature (word or n-gram) an explicit, individually inspectable score, so that any classification can be traced back to the specific lexical or contextual features that produced it, at the scale required for routine auditing rather than one-off analysis.

Computational efficiency is a secondary benefit of this design, not the primary motivation, since modern hardware makes Transformer inference broadly affordable; the design constraint we prioritize is auditable interpretability, and establishing how far a fully interpretable, sparsity-aware framework can be pushed under class imbalance is itself a necessary baseline against which future, less interpretable approaches can be compared. Because efficiency is not our central claim, we do not report standalone runtime or memory figures for our models in this study: a wall-clock or memory number is only informative relative to a comparable Transformer baseline trained and served under matched conditions, and reporting it in isolation would not support any claim we make here. A controlled head-to-head comparison against a fine-tuned Transformer, on both predictive performance and resource cost, is left as future work.

1.2.Problem Definition

Building upon our earlier work, this study formulates bias detection as an empirical investigation into the effects of contextual complexity under class-imbalanced conditions. Rather than assuming that richer linguistic context improves performance, we explicitly examine how different n-gram representations interact with sparsity, probability estimation, and minority-class detection.

The central objective of this work is to evaluate how increasing contextual and syntactic granularity in n-gram representations affects the reliability of bias detection in social media text. In particular, we analyze whether higher-order n-grams provide additional discriminatory power beyond unigram and bigram lexical features, or whether the resulting feature sparsity degrades minority-class performance under class imbalance. We further study the impact of incorporating part-of-speech information into trigram representations, and assess how a probabilistic log-likelihood-based scoring framework behaves across n-gram configurations with varying sparsity characteristics.

2.METHOD

This section first outlines the Z-Score Statistical Scoring baseline from our previous work [20], which serves as a baseline for bias detection. The proposed implementation, described in the subsequent subsections, is organized into three main components: n-gram preprocessing, log-likelihood estimation, and contextual n-gram based probabilistic bias detection. Each of these components is detailed below.

2.1.Z-Score Statistical Scoring

Algorithm 1 Bias Detection via Statistical Scoring and Threshold Optimization

---

1: Input:
2: Training dataset D_{train}
3: Test dataset D_{test}
4: WordNet Hashtable H_w
5: Threshold set $\mathcal{T}$
6: Output:
7: Optimal threshold $\hat{t}$
8: Evaluation metrics: Accuracy, F1 Score, Precision, Recall, Confusion Matrix, ROC Curve
9: Step 1: Preprocess and compute word frequencies
10: Tokenize, lowercase, and lemmatize all words in D_{train}
11: Remove stopwords and update frequency dictionary F
12: Step 2: Compute bias ratios
13: For each label y∈0,1. compute separate word frequency dictionaries
14: Use lemmatized words present in H_w
15: Compute ratio $R(w'_j) = \frac{C_{\text{bias}}(w'_j)}{C_{\text{non-bias}}(w'_j)}$
16: Sort the resulting ratio dictionary R in descending order
17: Step 3: Normalize ratios and compute Z-scores
18: Apply Min-Max normalization on $R \rightarrow R_{\text{norm}}$
19: Compute mean μ and std. dev $\sigma\ of\ (R_{\text{norm}})$
20: Calculate z-score: $Z(w'_j) = \frac{R_{\text{norm}}(w'_j) - \mu}{\sigma}$
21: Step 4: Calculate Bayesian word scores
22: For each $w'_j \in Z$, compute $b(w'_j) = z(w'_j) \times \frac{P_{\text{biased}}}{P(w'_j)}$
23: Store in Bayesian score dictionary b
24: Step 5: Compute tweet-level scores

25: For each tweet $t_i \in D_{test}$, lowercase, tokenize, and sum $b(w'_j)$
26: Store in tweet score dictionary t
27: Step 6: Evaluate thresholds and select optimum
28: For each threshold $\theta \in \mathcal{T}$:
29: Label tweet Ti as biased or not based on $t(T_i) \geq \theta$
30: Compute Accuracy, Precision, Recall, F1 Score, and Confusion Matrix
31: Update $\hat{t}$ if F1 and Accuracy improve
32: Step 7: Visualize and return results
33: Plot ROC Curve and Accuracy vs Threshold graph
34: **return** $\hat{t}$, Accuracy, F1 Score, Precision, Recall, Confusion Matrix, ROC Curve

The Z-Score Statistical Scoring algorithm (Algorithm 1) is structured as a sequence of statistical operations designed to extract lexical indicators of bias from social media text. This process starts with a comprehensive preprocessing step: all the tokens in the training and test sets are converted to lowercase and stopwords are removed.

Common uninformative words are removed, and lemmatization is performed to reduce words to their base forms. This normalization will make the following frequency analysis not be influenced by superficial differences in word forms.

The algorithm preprocesses the data and then creates a frequency dictionary for each class label which is biased and non-biased by calculating the frequency of each unique word in each of the sets of training data. The idea is that some words will be more likely to occur in biased contexts than non-biased ones and that the differences in these can be used as indicators of bias detection. A bias ratio is computed for each word using the ratio of the word's frequency in the biased class to its frequency in the non-biased class. The ratios are normalized by min-max scaling and then standardized by z-scoring for meaningful comparison across the vocabulary. This process emphasizes words that are more indicative of bias and diminishes the influence of common or neutral words.

These word-level z-scores are then combined to yield a bias score for each test instance. The z-scores of the words in each tweet are added together, and the z-scores for each word are multiplied by the relative frequency of the word in the corpus. This aggregation is meant to provide a representation of association with bias and general prevalence. The algorithm uses a series of possible thresholds, and labels tweets as biased if their scores are above the threshold. The threshold is determined by maximizing evaluation metrics like accuracy, F1-score etc., and it is empirically determined.

While this approach is clear and interpretable, it exhibits significant limitations when applied to n-gram features. Throughout this paper, we use the term sparsity to refer to the fact that as the window size of a feature grows (from unigrams to trigrams to pentagrams), the number of distinct feature combinations grows combinatorially while the corpus size remains fixed; consequently, longer word sequences appear very infrequently, often only once, making it difficult to estimate reliable statistics for them within either class.

This sparsity is precisely what breaks the Z-score aggregation described above. Step 3 of Algorithm 1 requires computing the standard deviation σ of the normalized bias ratios R_{norm} across the vocabulary. When most n-gram features occur only once or twice, their raw ratios $R(w'_j)$ collapse to a small number of degenerate values (typically 0, 1, or undefined when $C_{\text{non-bias}}(w'_j) = 0$), so Rnorm clusters tightly around a small set of points. This drives σ toward zero, and the resulting Z-scores $Z(w'_j) = (R_{\text{norm}}(w'_j) - \mu)/\sigma$ become numerically unstable or artificially inflated, no longer reflecting a meaningful degree of statistical deviation from the mean.

A second, independent failure mode arises one step later in the pipeline, at Step 4, and is not resolved even when Step 3's σ-collapse is controlled for (for instance, by restricting the vocabulary to features with a minimum occurrence count). Step 4 converts each word-level z-score into a final bias weight via

$$b(w'_j) = z(w'_j) \times \frac{P_{\text{biased}}}{P(w'_j)} \qquad (1)$$

where $P(w'_j)$ is the word's overall relative frequency in the training corpus, i.e., its total count across both classes divided by the total token count. This term is intended to downweight common, low-information words and upweight rare, potentially informative ones, which is a reasonable goal in isolation. However, because $P(w'_j)$ appears in the denominator, its effect is unbounded: as $P(w'_j) \to 0$, the multiplier $P_{\text{biased}}/P(w'_j) \to \infty$, regardless of how large or small zw'j itself is. In a vocabulary of several thousand distinct lemmatized words drawn from a corpus of only a few thousand training instances, as is typical for the datasets in this study, the great majority of words occur only a handful of times each, so $P(w'_j)$ is frequently on the order of 10^{-3} to 10^{-4} or smaller. Dividing by a value in this range amplifies $z(w'_j)$ by a factor of 10^3 to 10^4, which is sufficient on its own to produce the extreme, multi-digit bias weights observed empirically (Section 3.9.), independent of whether σ in Step 3 has already collapsed.

Because the $1/P(w'_j)$ amplification is a property of the scoring formula itself rather than a symptom of insufficient filtering, it persists even for features with moderate support: a word occurring nine or ten times in a training set of several thousand instances still has a small enough $P(w'_j)$ to be amplified by three to four orders of magnitude, well past the point where a minimum-count filter of the kind used elsewhere in this paper (Section 3.9) would exclude it. This is a materially different failure than the σ-collapse in Step 3, which is a consequence of sparse, degenerate ratio values and can in principle be mitigated by excluding low-count features before computing μ and σ; the Step 4 amplification cannot be fixed the same way, since it is structural to dividing by an empirical word frequency rather than a symptom of insufficient data.

Both failure modes identified above make it difficult for the Z-Score Baseline to handle the increased sparsity and complexity of n-gram representations: when extended to trigrams, pentagrams, or other higher-order features, the algorithm struggles to incorporate the richer contextual and syntactic information that these features provide, resulting in stagnant or even reduced performance compared to the unigram baseline. Both failure modes also motivate the same underlying fix: the log-likelihood ratio $\log\left(P_{\text{bias}}(w_j)/P_{\text{non-bias}}(w_j)\right)$ adopted in the remainder of this paper is, by construction, bounded for any fixed vocabulary size V once Laplace smoothing is applied (Section 2.), since each smoothed probability is bounded below by $1/(N+V) > 0$ and above by $(N+1)/(N+V) < 1$; the resulting log-ratio therefore cannot diverge to arbitrarily large magnitudes the way an unbounded division by $P(w'_j)$ can, regardless of how rare a given feature is. This limitation prompted the adoption of a probabilistic framework based on log-likelihood estimation with Laplacian smoothing, as described in the following subsections.

2.2.N-gram Feature Encoding with POS Tagging

To systematically explore the influence of contextual and syntactic granularity on bias detection, we implemented a flexible preprocessing pipeline (Figure 1) supporting four distinct feature extraction strategies: unigram (NO_Trigram), trigram without POS tags (TRIG_NOPOS), trigram with POS tags (TRIG), and pentagram with extended context (PENTA). For each strategy, the choice is carefully made to focus on one aspect of linguistic information so that a comparison can be made with the specific contribution to bias detection.

For each strategy, the encoding procedure is made formal in Algorithm 2. The unigram (NO_Trigram) model is a lexical baseline that records the occurrence of specific words without taking into account the local context. The TRIG_NOPOS strategy adds short-range context by learning contiguous three-word sequences, which helps to capture common collocations and phrase-level patterns that can be subtle indicators of bias. The TRIG strategy uses part-of-speech information to augment each trigram with syntactic tags so that the model can differentiate between the use of the same word as an adjective and as a noun. This syntactic granularity is proposed to improve the model's power to distinguish between seemingly alike phrases with different connotations as a function of their syntactic usage. The PENTA strategy additionally expands the context window to 5 tokens to see if longer sequences can identify more complicated and multi-word bias patterns that aren't seen in shorter n-grams.

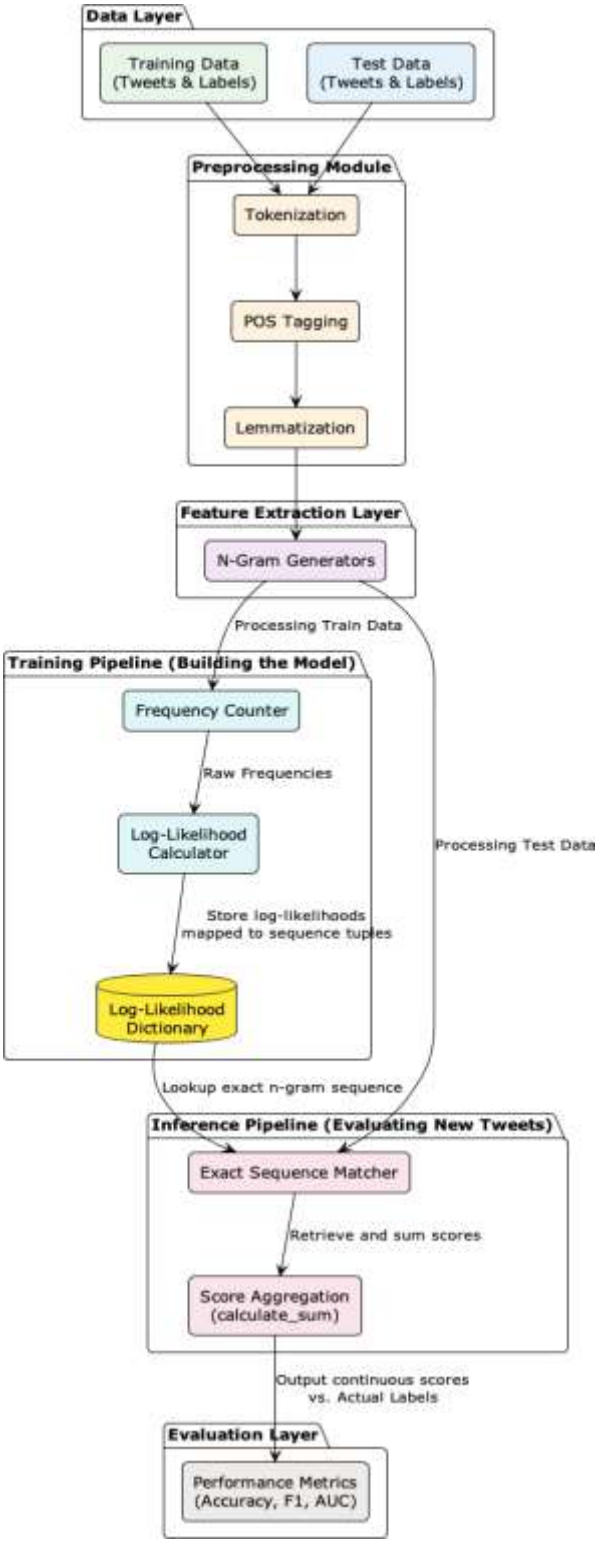


Figure 1. System architecture for contextual n-gram bias detection implementation.

We can then empirically evaluate the model expressiveness, the sparsity of the features used and their interpretability by varying the window size and by adding more complex linguistic features to simpler baseline models, and see whether the use of richer linguistic features results in better performance or not, and if so, whether it is worth the complexity.

Table 1. Illustration of n-gram encoding strategies for the example sentence

Strategy	Encoded Features
NO_Trigram	STR, The, doctor, prescribed, antibiotics, END
TRIG_NOPOS	[STR, The, doctor], [The, doctor, prescribed], [doctor, prescribed, antibiotics], [prescribed, antibiotics, END]
TRIG	(STR, The, NN), (DT, doctor, VBD), (NN, prescribed, NNS), (VBD, antibiotics, END)
PENTA	(BEG, BEG, The, NN, VBD), (BEG, DT, doctor, VBD, NNS), (DT, NN, prescribed, NNS, END), (NN, VBD, antibiotics, END, END)

To illustrate the effect of each encoding strategy, consider the sentence:
"The doctor prescribed antibiotics."
The width of the padding depends on the window size of the selected strategy. For NO_Trigram, TRIG_NOPOS, and TRIG (window radius 1), a single boundary token is added on each side, and the sentence becomes:
"STR The doctor prescribed antibiotics END"
For PENTA (window radius 2), two boundary tokens are required on each side so that every real token has a full five-token neighbourhood, and the padded sequence instead becomes:
"BEG BEG The doctor prescribed antibiotics END END"
We use STR/END to denote single-token boundaries and BEG/END to denote the doubled boundaries required by the wider pentagram window, consistent with the notation used in Table 1.
A trigram/pentagram window is applied, for illustration, a trigram window applied to the padded sequence yields:

- [STR The doctor]
- [The doctor prescribed]
- [doctor prescribed antibiotics]
- [prescribed antibiotics END]

Further, if the POS-augmented strategy is selected, the tokens are replaced with their corresponding POS tags (e.g., DT, NN, VBD), producing:

- (STR,The,NN)
- (DT,doctor,VBD)
- (NN,prescribed,NNS)
- (VBD,antibiotics,END)

Each of the resulting core words in the n-grams is then checked against the WordNet-based hash table HW . This filtering step enforces standardization for subsequent analysis. In our earlier work [9], WordNet filtering was applied to single words to remove noise from a unigram vocabulary; extending the same filter to the core word of a trigram or pentagram is a stronger constraint, since it also discards any n-gram whose central token is slang, a misspelling, or a proper noun, categories that are often themselves informative for bias detection on social media text. We retain WordNet filtering for n-grams nonetheless, treating it as a conservative noise-reduction step: it keeps the vocabulary anchored to well-formed lexical items so that sparsity is driven by genuine contextual variation in the surrounding window rather than by typographical noise, at the acknowledged cost of discarding some slang- or entity-bearing bias signals. Table 1 demonstrates how this sentence is transformed under each of the four preprocessing schemes.

Algorithm 2 N-Gram Encoding Strategy with POS

1: **Input:**
2: Text x_i
3: WordNet Hashtable H_W
4: Strategy flag $s \in \{\texttt{TRIG}, \texttt{TRIG\_NOPOS}, \texttt{NO\_Trigram}, \texttt{PENTA}\}$
5: **Output:**
6: Processed text Wi
7: Initialize POS tagger function: $\texttt{POS}(x)$: $x \mapsto \{\texttt{NNP}, \texttt{VBZ}, \texttt{VBG}, \texttt{IN}, \texttt{NN}, \texttt{CD}, \$\}$
8: Preprocess text:
9: **if** s = PENTA **then**
10: Prepend/append tokens: $T_i \leftarrow [\texttt{BEG}, \texttt{BEG}] \parallel x_i \parallel [\texttt{END}, \texttt{END}]$
11: **else**
12: Prepend/append tokens: $T_i \leftarrow [\texttt{STR}] \parallel x_i \parallel [\texttt{END}]$

13: **end if**
14: Initialize empty word list: $\mathcal{W}_i \leftarrow \emptyset$
15: **if** $s \in$ {TRIG, PENTA} then
16: for each token $w_k \in T_i$ do
17: Assign POS tag: : $pos_k \leftarrow$ POS(w_k)
18: **end for**
19: **end if**
20: **switch** (s)
21: **case** TRIG:
22: **for** $k = 2$ *to* $|T_i| - 1$ **do**
23: Form trigram: w ← (posk−1, wk, posk+1)
24: **if** $w \in H_W$ **then**
25: $\mathcal{W}_i \leftarrow \mathcal{W}_i \cup \{w\}$
26: **end if**
27: **end for**
28: **case** TRIG_NOPOS:
29: **for** $k = 2$ *to* $|T_i| - 1$ **do**
30: Form trigram: $w \leftarrow (pos_{k-1}, w_k, pos_{k+1})$
31: **if** $w \in H_W$ **then**
32: $\mathcal{W}_i \leftarrow \mathcal{W}_i \cup \{w\}$
33: **end if**
34: **end for**
35: **case** NO_Trigram:
36: **for** each token $w_k \in T_i$ **do**
37: **if** $w \in H_W$ **then**
38: $\mathcal{W}_i \leftarrow \mathcal{W}_i \cup \{w\}$
39: **end if**
40: **end for**
41: **case** PENTA:
42: **for** $k = 3$ *to* $|T_i| - 2$ **do**
43: Form pentagram:
44: $w \leftarrow (pos_{k-2}, pos_{k-1}, w_k, pos_{k+1}, pos_{k+2})$
45: **if** $w \in H_W$ **then**
46: $\mathcal{W}_i \leftarrow \mathcal{W}_i \cup \{w\}$
47: **end if**
48: **end for**
49: **end switch**
50: **return** W_i

2.3.Loglikelihood based Bias Score Computation

Building on the limitations of frequency-based and z-score approaches discussed earlier, we adopt a log-likelihood estimation framework to quantify the association between linguistic features and bias. Algorithm 3 outlines the stepwise procedure for computing these feature-level bias scores.

The process begins by aggregating frequency counts for each feature whether a word or an n-gram across both the biased and non-biased classes. For each feature w_j, the frequency dictionary F records its occurrences in both classes, denoted as $C_{bias}(w_j)$ and $C_{non\text{-}bias}(w_j)$. Summing these counts across the vocabulary yields the total number of biased and non-biased tokens, N_{bias} and $N_{non\text{-}bias}$, as well as the overall vocabulary size V.

Next, the algorithm estimates the prior probabilities of each class from the training data, as shown in Equation 2,

$$P(\text{bias}) = \frac{N_{\text{bias}}}{N_{\text{total}}}, \qquad P(\text{non-bias}) = \frac{N_{\text{non-bias}}}{N_{\text{total}}} \qquad (2)$$

where N_{total} is the sum of all tokens, given by

$$N_{\text{total}} = N_{\text{bias}} + N_{\text{non-bias}} \qquad (3)$$

Algorithm 3 Bias Score Computation via Log-Likelihood Estimation

1: Input:
2: Frequency dictionary F, where for each word or phrase w_j,
3: $F(w_j) = \left(C_{\text{non-bias}}(w_j), C_{\text{bias}}(w_j)\right)$
4: Output:

5: Bias score dictionary B, mapping each wj to its log-likelihood bias score
6: Step 1: Aggregate corpus-level counts
7: Compute N_{bias}, $N_{non\text{-}bias}$, N_{total}, and vocabulary size V = |F |
8: Step 2: Compute prior bias ratio
9: Estimate class priors P (bias), P (non-bias) from training counts (Eq. 2)
10: Compute log-prior ratio $log\ P_{prior}$
11: Step 3: Estimate smoothed likelihoods and compute bias scores
12: Initialize B←∅
13: **for** each word or n-gram $w_j \in F$ **do**
14: Compute Laplace-smoothed conditionals (Eqs. 4, 5)
15: Compute log-likelihood ratio score (Eq. 6)
16: $\mathcal{B}[w_j]$←score
17: **end for**
18: **Step 4: Return result**
19: **return** bias score dictionary B

The log-ratio of these priors, log P_{prior}, provides a global reference for interpreting feature associations. To ensure robust probability estimates, especially for rare or unseen features, Laplace (add-one) smoothing is applied. The smoothed conditional probabilities for each feature w_j are computed as shown in Equations 4 and 5:

$$P_{\text{bias}}(w_j) = \frac{C_{\text{bias}}(w_j)+1}{N_{\text{bias}}+V} \quad (4)$$

$$P_{\text{non-bias}}(w_j) = \frac{C_{\text{non-bias}}(w_j)+1}{N_{\text{non-bias}}+V} \quad (5)$$

This smoothing prevents zero probabilities and ensures that every feature contributes meaningfully to the final score.
The core of the algorithm is the calculation of the log-likelihood ratio for each feature, as defined in Equation 6:

$$\text{Score}(w_j) = \log\left(\frac{P_{\text{bias}}(w_j)}{P_{\text{non-bias}}(w_j)}\right) \quad (6)$$

This value measures the amount of bias that the w_j value indicates. Items with positive scores are more likely to be associated with biased content, and those with negative scores are more likely to be associated with non-biased language. The resulting bias score dictionary is then used as a principled mapping from features to their estimated bias association, which is used to score and classify tweets in a downstream step.
The proposed algorithm for contextual n-gram bias detection (Algorithm 4) directly extends the original statistical scoring framework with two major updates: (1) a flexible, context-aware, feature extraction pipeline, and (2) a log-likelihood-based bias scoring mechanism based directly on class-conditional probabilities instead of ad hoc normalization. For each instance, the pipeline preprocesses the instance with the n-gram encoding method chosen, creates contextually relevant features, and calculates the log-likelihood bias scores by applying Laplace smoothing to the class-conditional probabilities. For all test instances, the model puts them together to get a tweet-level bias score.

Algorithm 4 Contextual N-Gram Bias Detection Pipeline (Proposed Work)

1: **Input:**
2: Training dataset D_{train}
3: Test dataset D_{test}
4: WordNet Hashtable H_W
5: n-gram strategy flag s
6: Threshold set $\mathcal{T}$
7: **Output:**
8: Optimal threshold $\hat{t}$
9: Evaluation metrics: Accuracy, F1 Score, Precision, Recall, Confusion Matrix, ROC Curve
10: **Step 1: Preprocess and extract features**
11: For each sample in D_{train}:
12: Lowercase, remove stopwords, and lemmatize words
13: Apply n-gram encoding (with or without POS tags) according to strategy s
14: Update frequency dictionary F by class label
15: **Step 2: Compute feature probabilities with Laplace smoothing**

16: For each feature in F :
17: Compute class-conditional probabilities for each class (with Laplace smoothing)
18: **Step 3: Calculate log-likelihood bias scores**
19: For each feature:
20: Compute log-likelihood ratio and store in bias score dictionary B
21: **Step 4: Compute tweet-level scores**
22: For each sample $t_i \in D_{\text{test}}$:
23: Preprocess and encode as in Step 1
24: Sum log-likelihood bias scores for all features in t_i
25: Store in tweet score dictionary t
26: **Step 5: Evaluate thresholds and select optimum**
27: For each threshold $\theta \in T$
28: For each tweet t_i:
29: Assign label: biased if $t(t_i) \geq \theta$, else non-biased
30: Compute evaluation metrics (Accuracy, Precision, Recall, F1, etc.)
31: Update $\hat{t}$ if Accuracy or F1 Score metrics improve
32: **Step 6: Visualize and return results**
33: Plot ROC Curve and Accuracy vs Threshold graph
34: return $\hat{t}$, Accuracy, F1 Score, Precision, Recall, Confusion Matrix, ROC Curve

---

Classification is performed by evaluating a range of possible thresholds on these tweet-level scores, selecting the threshold that optimizes F1-score or accuracy on the validation set. Comparing Algorithm 1 with Algorithm 4, this unified approach extends directly to different n-gram configurations through the strategy flag s, and consolidates the separate normalization, z-score, and Bayesian-weighting steps of the original pipeline into a single Laplace-smoothed log-likelihood computation applied identically across all five feature-extraction strategies.

3.RESULTS AND DISCUSSION

The implementation is carried out using Python 3.12, leveraging NLTK's [21] WordNet [22] for hash table generation. For part-of-speech (POS) tagging, the spaCy library [23] is utilized, specifically loading the en_core_web_sm model. The three primary datasets from our previous work are retained, with an additional new dataset introduced to evaluate the performance of the algorithms on large-scale datasets exhibiting skewed representations.

Table 2. Summary of Datasets Used in Simulations

ID	Dataset	Description	Records	Data Distribution
1	TwitterDF Dataset	Hate tweets consisting of racial and gender discrimination	20989	Biased: 30%, Unbiased:70%
2	Labeled Data	Tweets classified into hate-speech, offensive language, neither	24783	Hate-Speech: 1430, Offensive: 19190, Neither: 4163
3	Political Social Media Posts	Social media posts from various political entities	5000	Neutral: 74%, Partisan: 26%
4	Kaggle Jigsaw Dataset	Annotated data of the Civil comments platform	199516	Toxic: 8.00%, Non toxic: 92.00%

3.1.Datasets Used

Table 2 gives an overview of the datasets used. Dataset 1 consists of 20,989 tweets compiled from multiple publicly available sources [24–26] and focuses on hate speech related to racial and gender-based discrimination. Within this dataset, 30% of the tweets are labeled as biased, while the remaining 70% are marked as unbiased. Dataset 2 includes 24,783 tweets sourced from Twitter [27], categorized into three classes: hate speech (5%), offensive language (77%), and neutral (16%). The hate speech and offensive language classes were combined into one "offensive" class because of the binary nature of our classifier. As a broad concept, for the purposes of this empirical study, we include hate speech, toxicity, offensive language, and the partisan and discriminatory content found in Datasets 1, 3, and 4 under the umbrella of bias. These constructs are not exactly the same, but in each case, their language is presented as a binary classification: neutral, in-group-preserving versus disparaging, discriminatory, or hostile toward a target,

the shared property we are interested in our classifier learning to detect. The 5000 posts in dataset 3 [28] are from social media, written by political actors, and 74% of which are neutral, 26% partisan. As such Dataset 4 was designed intentionally as a large-scale real-world set of data, not a carefully curated benchmark, to test model performance in situations that mimic real-world use. The Jigsaw Civil Comments data [29] is a naturally occurring class skew (about 8% toxic), without any preprocessing or balancing, and is representative of the extreme imbalance a production bias detection system would face. This is different from the first three data sets and examines a qualitatively different type of failure mode – majority-class dominance with extreme skew – as seen in Table 2.

3.2.Performance Analysis

We employ nine evaluation metrics; Accuracy, Balanced Accuracy, Precision, Recall, Specificity, G-Mean, F1 Score, ROC AUC, and Confusion Matrix, to capture different aspects of classifier performance under class imbalance, as summarised in Table 3. Because Accuracy alone is known to be misleading on imbalanced data (a classifier that always predicts the majority class can score highly while never identifying a single minority-class instance), we report Balanced Accuracy and G-Mean alongside it throughout; the two are related but not interchangeable, since G-Mean's multiplicative form drives it to exactly zero whenever either class is completely misclassified, whereas Balanced Accuracy's additive form does not.

Each classifier's decision threshold can be tuned toward different objectives, and different objectives can select materially different thresholds on the same trained model. To make this explicit rather than reporting a single, silently chosen threshold, we report every metric under three separate threshold-selection regimes: a threshold that maximizes Accuracy, one that maximizes F1 Score, and one that maximizes ROC AUC evaluated on binarized predictions at that threshold. Every metric reported under a given regime, including the confusion matrix, is computed at that regime's single threshold, so no two numbers in the same regime can describe two different classifiers. ROC AUC itself is the one exception: because it is computed directly on the continuous, unthresholded scores, it does not vary by regime, and we report it once, separately, in Table 7.

We implemented and compared five feature extraction approaches, Z-Score Baseline, NO Trigram, Trigram, Trigram NO POS, and Pentagram, on their ability to detect biased or toxic content across four datasets. Since these datasets exhibit varying degrees of class imbalance, model performance must be interpreted in that context: minority-class metrics such as Recall, F1 Score, and G-Mean provide more meaningful insight than overall Accuracy, which can be dominated by majority-class predictions. We focus on G-Mean and Balanced Accuracy in addition to F1 because in this context the main concern is reliable identification of minority-class examples (biased, offensive or toxic), rather than only on Accuracy in isolation. The proposed models are compared to the previous work [9] and improvements over [9] are considered improvements over those previous methods when the datasets and evaluation conditions are comparable. We now look at each of the three threshold regimes in turn, followed by the results that each will yield, and then consider the threshold-independent overall ROC AUC and the corresponding ROC curves.

Table 3. Evaluation metrics used in simulations

Metric	Description	Equation
Accuracy	Fraction of correct predictions over all instances	$\frac{TP + TN}{TP + TN + FP + FN}$
Balanced Accuracy	Arithmetic mean of sensitivity and specificity; corrects Accuracy's bias toward the majority class under imbalance	$\frac{\text{Sensitivity} + \text{Specificity}}{2}$
Precision	Fraction of true positives among all positive predictions	$\frac{TP}{TP + FP}$
Recall (Sensitivity)	Fraction of actual positives correctly identified	$\frac{TP}{TP + FN}$
Specificity	Fraction of actual negatives correctly identified	$\frac{TN}{TN + FP}$
G-Mean	Geometric mean of sensitivity and specificity; unlike Balanced Accuracy, collapses to zero if either class is completely misclassified, making it a stricter imbalance-aware metric	$\sqrt{\text{Sensitivity} \times \text{Specificity}}$

F1 Score	Harmonic mean of precision and recall	$2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$
Confusion Matrix	$$2\times2$$ count table of prediction outcomes	$\begin{bmatrix} TN & FP \\ FN & TP \end{bmatrix}$
ROC AUC	Area under the ROC curve, computed on continuous scores; measures discrimination independent of any classification threshold	

Table 4. Metrics at the Accuracy-Optimized Threshold

Approach	Metrics	1st Dataset	2nd Dataset	3rd Dataset	4th Dataset
Z-Score	Accuracy	82.09%	83.99%	74.40%	92.01%
	Balanced Accuracy	0.74	0.53	0.53	0.50
	Precision	0.78	0.84	0.59	0.00
	Recall	0.55	1.00	0.08	0.00
	Specificity	0.93	0.05	0.98	1.00
	G-Mean	0.72	0.23	0.27	0.00
	F1 Score	0.65	0.91	0.14	0.00
	Confusion Matrix	$\begin{bmatrix} 1376 & 96 \\ 280 & 347 \end{bmatrix}$	$\begin{bmatrix} 22 & 394 \\ 3 & 2060 \end{bmatrix}$	$\begin{bmatrix} 362 & 7 \\ 121 & 10 \end{bmatrix}$	$\begin{bmatrix} 183974 & 0 \\ 15978 & 0 \end{bmatrix}$
Trigram NO POS	Accuracy	78.04%	84.99%	74.20%	92.24%
	Balanced Accuracy	0.65	0.58	0.51	0.53
	Precision	0.85	0.86	0.67	0.63
	Recall	0.32	0.98	0.03	0.07
	Specificity	0.98	0.19	0.99	1.00
	G-Mean	0.56	0.43	0.17	0.27
	F1 Score	0.47	0.92	0.06	0.13
	Confusion Matrix	$\begin{bmatrix} 1436 & 36 \\ 425 & 202 \end{bmatrix}$	$\begin{bmatrix} 77 & 339 \\ 33 & 2030 \end{bmatrix}$	$\begin{bmatrix} 367 & 2 \\ 127 & 4 \end{bmatrix}$	$\begin{bmatrix} 183297 & 677 \\ 14841 & 1137 \end{bmatrix}$
NO Trigram	Accuracy	90.81%	89.96%	77.00%	92.47%
	Balanced Accuracy	0.89	0.79	0.63	0.60
	Precision	0.85	0.93	0.61	0.58
	Recall	0.83	0.96	0.35	0.21
	Specificity	0.94	0.62	0.92	0.99
	G-Mean	0.89	0.77	0.57	0.46
	F1 Score	0.84	0.94	0.44	0.31
	Confusion Matrix	$\begin{bmatrix} 1383 & 89 \\ 104 & 523 \end{bmatrix}$	$\begin{bmatrix} 257 & 159 \\ 90 & 1973 \end{bmatrix}$	$\begin{bmatrix} 339 & 30 \\ 85 & 46 \end{bmatrix}$	$\begin{bmatrix} 181511 & 2463 \\ 12586 & 3392 \end{bmatrix}$
Trigram	Accuracy	90.61%	88.18%	76.00%	92.83%
	Balanced Accuracy	0.88	0.71	0.62	0.61
	Precision	0.87	0.90	0.57	0.64
	Recall	0.81	0.97	0.34	0.23
	Specificity	0.95	0.45	0.91	0.99
	G-Mean	0.87	0.66	0.55	0.48
	F1 Score	0.84	0.93	0.42	0.34
	Confusion Matrix	$\begin{bmatrix} 1396 & 76 \\ 121 & 506 \end{bmatrix}$	$\begin{bmatrix} 187 & 229 \\ 64 & 1999 \end{bmatrix}$	$\begin{bmatrix} 336 & 33 \\ 87 & 44 \end{bmatrix}$	$\begin{bmatrix} 181861 & 2113 \\ 12229 & 3749 \end{bmatrix}$
Pentagram	Accuracy	87.23%	84.23%	75.20%	92.17%

	Balanced Accuracy	0.84	0.56	0.54	0.52
	Precision	0.81	0.85	0.67	0.75
	Recall	0.75	0.98	0.11	0.03
	Specificity	0.92	0.14	0.98	1.00
	G-Mean	0.83	0.37	0.32	0.18
	F1 Score	0.78	0.91	0.18	0.06
	Confusion Matrix	$\begin{bmatrix}1360 & 112\\ 156 & 471\end{bmatrix}$	$\begin{bmatrix}58 & 358\\ 33 & 2030\end{bmatrix}$	$\begin{bmatrix}362 & 7\\ 117 & 14\end{bmatrix}$	$\begin{bmatrix}183805 & 169\\ 15479 & 499\end{bmatrix}$

3.3. Accuracy-Optimized Threshold

Dataset 4 represents an extreme class-imbalance scenario, with toxic instances comprising roughly 8% of the corpus; under such conditions, feature-based classifiers are prone to majority-class dominance, and the threshold regime chosen matters enormously. The Z-Score Baseline gives the clearest illustration: when its threshold is tuned to maximize Accuracy, it converges on the trivial classifier that predicts "non-toxic" for every single instance (TN 183,974, TP 0, FN 15,978), yielding 92.01% accuracy while never once correctly identifying a toxic comment; G-Mean for this threshold is exactly 0, which is precisely the failure mode G-Mean is designed to expose and Accuracy is not.

This is not an isolated artifact of the Z-Score model: on Dataset 4, all five approaches converge to Accuracy within a narrow 92.01–92.83% band under this regime, yet Recall never exceeds 0.23 for any of them (Z-Score 0.00, Pentagram 0.03, Trigram NO POS 0.07, NO Trigram 0.21, Trigram 0.23) and G-Mean ranges from 0.00 to only 0.48. An accuracy-maximizing search has every incentive to settle near this operating point once toxic comments fall to 8% of the corpus, since correctly labelling a handful of additional majority-class instances outweighs, in raw accuracy terms, correctly identifying several toxic ones. The effect is weaker but still visible on Dataset 3, the least skewed of the four (74%/26%): Accuracy across the five approaches varies by less than three points (74.20–77.00%), while G-Mean spans a much wider 0.17–0.57, showing that even modest imbalance is enough to decouple the two metrics once the threshold is chosen for Accuracy alone. Within this regime, NO Trigram and Trigram remain the most consistent pair, tracking each other within one to two points of Balanced Accuracy and G-Mean on every dataset, while Trigram NO POS and Pentagram fall furthest behind specifically on Datasets 3 and 4, where their higher-order, sparser features leave too few reliably estimated positive-class scores for the accuracy search to locate a threshold that catches minority-class instances at all.

3.4.F1-Optimized Threshold

Retuning the same trained model with the Z-Score to maximize F1 recovers some minority class sensitivity (Recall 0.10, Precision 0.35, G-Mean 0.32) at a small accuracy cost (91.27%), which suggests that the performance drop was not caused by the Z-Score scores themselves, but rather by a threshold selection problem on top of a weak underlying model. Pentagram has a similar, but less severe, behaviour at this threshold (Recall 0.26, Specificity 0.86, G-Mean 0.47): the model does not know enough about the high order features to separate the classes cleanly at any threshold, so it adapts to a compromise position that is not catastrophic by any measure, but rather, is mediocre by all measures. With F1-optimized threshold, the range of NO Trigram's F1 is 0.47 to 0.94, and its G-Mean ranges from 0.67 to 0.89 across the four datasets, demonstrating that simple unigram lexical representation captures the core bias signals while maintaining reasonable specificity.

The example of the Trigram NO POS variant shows that a classifier can be summarized inaccurately by F1. Its F1 score at the F1 optimized threshold is promising on Dataset 2 (0.92) and poor on Dataset 4 (0.24) on its own, which may indicate that it does not transfer well from small to large datasets. As for G-Mean and Specificity, they tell a more telling story: On Dataset 2, Specificity is 0.19 (Recall 0.98) — the model is getting it right with the majority class almost randomly on a dataset in which the "biased" class is actually the numerical majority (83%). The pattern on Dataset 4 is reversed, with Specificity reaching 0.98 and Recall dropping to 0.17, so that the model is now favouring the majority (non-toxic) class. In both cases, G-Mean stays low (0.43 and 0.41 respectively) because it is not fooled by whichever class happens to dominate a given dataset; the model is not learning a stable, dataset-independent notion of "bias" so much as tracking each dataset's specific class balance.

3.5.ROC-AUC-Optimized Threshold

Table 6 reports the same five approaches under the threshold that maximizes ROC AUC evaluated on binarized predictions, providing a third reference point alongside the Accuracy- and F1-optimized regimes above.

Retuning toward this objective consistently pulls thresholds away from the majority-class-favoring optima that Accuracy-optimization finds, and the price paid in raw Accuracy scales with how sparse the

underlying features are. NO Trigram and Trigram absorb this retuning with only modest Accuracy loss: NO Trigram falls from 92.47% (Accuracy-optimized) to 84.37% on Dataset 4, but in exchange G-Mean rises from 0.46 to 0.78 and Recall from 0.21 to 0.70, indicating the model was already capable of much better-balanced detection that Accuracy-optimization had simply been hiding. Trigram NO POS, by contrast, is far more volatile: its Accuracy collapses from 78.04%/84.99%/74.20%/92.24% (Accuracy-optimized) to 53.03%/64.78%/72.40%/64.55% under this regime, a far steeper trade than any other approach exhibits, while its G-Mean gains are inconsistent across datasets (0.56 vs. 0.56 on Dataset 1, but 0.71 vs. 0.43 on Dataset 2 and 0.59 vs. 0.27 on Dataset 4). This volatility is itself informative: it suggests the Trigram NO POS score distribution is poorly separated, so that no single threshold delivers both good Accuracy and good balance simultaneously, unlike NO Trigram and Trigram, where a better-balanced operating point is available at comparatively little Accuracy cost.

Table 5. Metrics at the F1-Optimized Threshold

Approach	Metrics	1st Dataset	2nd Dataset	3rd Dataset	4th Dataset
Z-Score	Accuracy	81.75%	83.99%	69.20%	91.27%
	Balanced Accuracy	0.76	0.53	0.63	0.54
	Precision	0.73	0.84	0.42	0.35
	Recall	0.62	1.00	0.50	0.10
	Specificity	0.90	0.05	0.76	0.98
	G-Mean	0.75	0.23	0.61	0.32
	F1 Score	0.67	0.91	0.46	0.16
	Confusion Matrix	$\begin{bmatrix}1325 & 147\\ 236 & 391\end{bmatrix}$	$\begin{bmatrix}22 & 394\\ 3 & 2060\end{bmatrix}$	$\begin{bmatrix}281 & 88\\ 66 & 65\end{bmatrix}$	$\begin{bmatrix}180822 & 3152\\ 14304 & 1674\end{bmatrix}$
Trigram NO POS	Accuracy	53.03%	84.99%	35.40%	91.28%
	Balanced Accuracy	0.66	0.58	0.54	0.57
	Precision	0.39	0.86	0.28	0.39
	Recall	0.98	0.98	0.92	0.17
	Specificity	0.34	0.19	0.15	0.98
	G-Mean	0.58	0.43	0.38	0.41
	F1 Score	0.56	0.92	0.43	0.24
	Confusion Matrix	$\begin{bmatrix}497 & 975\\ 11 & 616\end{bmatrix}$	$\begin{bmatrix}77 & 339\\ 33 & 2030\end{bmatrix}$	$\begin{bmatrix}57 & 312\\ 11 & 120\end{bmatrix}$	$\begin{bmatrix}179792 & 4182\\ 13253 & 2725\end{bmatrix}$
NO Trigram	Accuracy	90.81%	89.87%	64.60%	90.85%
	Balanced Accuracy	0.89	0.76	0.67	0.73
	Precision	0.85	0.91	0.40	0.44
	Recall	0.83	0.97	0.71	0.51
	Specificity	0.94	0.54	0.62	0.94
	G-Mean	0.89	0.73	0.67	0.70
	F1 Score	0.84	0.94	0.51	0.47
	Confusion Matrix	$\begin{bmatrix}1383 & 89\\ 104 & 523\end{bmatrix}$	$\begin{bmatrix}226 & 190\\ 61 & 2002\end{bmatrix}$	$\begin{bmatrix}230 & 139\\ 38 & 93\end{bmatrix}$	$\begin{bmatrix}173444 & 10530\\ 7757 & 8221\end{bmatrix}$
Trigram	Accuracy	90.57%	88.18%	55.40%	91.72%
	Balanced Accuracy	0.89	0.71	0.64	0.67
	Precision	0.83	0.90	0.35	0.48
	Recall	0.86	0.97	0.82	0.37
	Specificity	0.92	0.45	0.46	0.96
	G-Mean	0.89	0.66	0.61	0.60
	F1 Score	0.85	0.93	0.49	0.42
	Confusion Matrix	$\begin{bmatrix}1361 & 111\\ 87 & 540\end{bmatrix}$	$\begin{bmatrix}187 & 229\\ 64 & 1999\end{bmatrix}$	$\begin{bmatrix}170 & 199\\ 24 & 107\end{bmatrix}$	$\begin{bmatrix}177527 & 6447\\ 10106 & 5872\end{bmatrix}$
Pentagram	Accuracy	86.99%	84.07%	50.00%	81.16%

	Balanced Accuracy	0.85	0.53	0.61	0.56
	Precision	0.77	0.84	0.33	0.14
	Recall	0.80	1.00	0.85	0.26
	Specificity	0.90	0.06	0.37	0.86
	G-Mean	0.85	0.24	0.57	0.47
	F1 Score	0.79	0.91	0.47	0.18
	Confusion Matrix	$\begin{bmatrix}1323 & 149\\124 & 503\end{bmatrix}$	$\begin{bmatrix}23 & 393\\2 & 2061\end{bmatrix}$	$\begin{bmatrix}138 & 231\\19 & 112\end{bmatrix}$	$\begin{bmatrix}158203 & 25771\\11892 & 4086\end{bmatrix}$

Table 6. Metrics at the ROC-AUC-Optimized Threshold

Approach	Metrics	1st Dataset	2nd Dataset	3rd Dataset	4th Dataset
Z-Score	Accuracy	81.75%	75.35%	69.20%	91.27%
	Balanced Accuracy	0.76	0.73	0.63	0.54
	Precision	0.73	0.93	0.42	0.35
	Recall	0.62	0.77	0.50	0.10
	Specificity	0.90	0.69	0.76	0.98
	G-Mean	0.75	0.73	0.61	0.32
	F1 Score	0.67	0.84	0.46	0.16
	Confusion Matrix	$\begin{bmatrix}1325 & 147\\236 & 391\end{bmatrix}$	$\begin{bmatrix}289 & 127\\484 & 1579\end{bmatrix}$	$\begin{bmatrix}281 & 88\\66 & 65\end{bmatrix}$	$\begin{bmatrix}180822 & 3152\\14304 & 1674\end{bmatrix}$
Trigram NO POS	Accuracy	53.03%	64.78%	72.40%	64.55%
	Balanced Accuracy	0.66	0.72	0.54	0.59
	Precision	0.39	0.95	0.43	0.12
	Recall	0.98	0.61	0.15	0.53
	Specificity	0.34	0.83	0.93	0.66
	G-Mean	0.58	0.71	0.38	0.59
	F1 Score	0.56	0.74	0.22	0.19
	Confusion Matrix	$\begin{bmatrix}497 & 975\\11 & 616\end{bmatrix}$	$\begin{bmatrix}344 & 72\\801 & 1262\end{bmatrix}$	$\begin{bmatrix}342 & 27\\111 & 20\end{bmatrix}$	$\begin{bmatrix}120637 & 63337\\7544 & 8434\end{bmatrix}$
NO Trigram	Accuracy	90.38%	83.22%	64.60%	84.37%
	Balanced Accuracy	0.89	0.87	0.67	0.78
	Precision	0.83	0.98	0.40	0.30
	Recall	0.85	0.81	0.71	0.70
	Specificity	0.93	0.92	0.62	0.86
	G-Mean	0.89	0.87	0.67	0.78
	F1 Score	0.84	0.89	0.51	0.42
	Confusion Matrix	$\begin{bmatrix}1365 & 107\\95 & 532\end{bmatrix}$	$\begin{bmatrix}382 & 34\\382 & 1681\end{bmatrix}$	$\begin{bmatrix}230 & 139\\38 & 93\end{bmatrix}$	$\begin{bmatrix}157442 & 26532\\4727 & 11251\end{bmatrix}$
Trigram	Accuracy	89.61%	83.26%	70.80%	85.16%
	Balanced Accuracy	0.90	0.81	0.65	0.70
	Precision	0.79	0.95	0.45	0.28
	Recall	0.90	0.84	0.53	0.52
	Specificity	0.90	0.79	0.77	0.88
	G-Mean	0.90	0.81	0.64	0.68
	F1 Score	0.84	0.89	0.49	0.36
	Confusion Matrix	$\begin{bmatrix}1319 & 153\\65 & 562\end{bmatrix}$	$\begin{bmatrix}328 & 88\\327 & 1736\end{bmatrix}$	$\begin{bmatrix}284 & 85\\61 & 70\end{bmatrix}$	$\begin{bmatrix}161923 & 22051\\7612 & 8366\end{bmatrix}$
Pentagram	Accuracy	85.09%	74.63%	50.00%	61.19%
	Balanced Accuracy	0.85	0.66	0.61	0.56

Precision	0.71	0.89	0.33	0.10
Recall	0.85	0.79	0.85	0.51
Specificity	0.85	0.54	0.37	0.62
G-Mean	0.85	0.65	0.57	0.56
F1 Score	0.77	0.84	0.47	0.17
Confusion Matrix	$\begin{bmatrix}1253 & 219\\ 94 & 533\end{bmatrix}$	$\begin{bmatrix}225 & 191\\ 438 & 1625\end{bmatrix}$	$\begin{bmatrix}138 & 231\\ 19 & 112\end{bmatrix}$	$\begin{bmatrix}114229 & 69745\\ 7855 & 8123\end{bmatrix}$

The Z-Score Baseline illustrates the same effect from the opposite direction. On Datasets 1, 3, and 4, its ROC-AUC-optimized threshold reproduces the identical confusion matrix obtained under F1-optimization, indicating the two objectives select the same operating point once Accuracy is no longer the sole criterion. On Dataset 2, however, the two regimes diverge sharply: F1-optimization leaves Z-Score at the near-trivial threshold that flags almost every tweet as offensive (Recall 1.00, Specificity 0.05, G-Mean 0.23, Accuracy 83.99%), because on this dataset the "offensive" class is itself the numerical majority (77%) and predicting it indiscriminately still yields a high F1 score; ROC-AUC-optimization instead selects a markedly more balanced threshold (Recall 0.77, Specificity 0.69, G-Mean 0.73) at a lower Accuracy of 75.35%. This is the clearest case in our results of F1 being fooled by class prevalence in exactly the way Section 3. anticipates, and of ROC-AUC-based threshold selection correcting for it. Pentagram shows the limits of what any threshold search can achieve when the underlying scores are weak: even at its best-balanced operating point, its G-Mean tops out at 0.32–0.57 across datasets and never approaches the 0.78–0.90 range NO Trigram and Trigram reach, confirming that its shortfall reflects genuinely limited class separability in its sparse, high-order feature scores rather than a poorly chosen threshold.

3.6.Comparison Across Threshold-Selection Regimes

Taken together, the three regimes show that the choice of threshold-selection objective is not a minor implementation detail but can change which model looks best and by how much. Under Accuracy-optimization, Dataset 4 makes all five approaches look nearly indistinguishable (92.01–92.83% Accuracy), masking the fact that their minority-class detection ability differs enormously (G-Mean 0.00–0.48). Switching to F1- or ROC-AUC-optimization removes this camouflage and separates NO Trigram and Trigram clearly from Z-Score, Trigram NO POS, and Pentagram on every dataset. This pattern holds across all four datasets, not only Dataset 4: the ranking of the five approaches by G-Mean or Balanced Accuracy is materially more stable across the F1- and ROC-AUC-optimized regimes than either is to the Accuracy-optimized regime, since Accuracy-optimization is the one most willing to sacrifice minority-class Recall entirely once doing so improves the raw fraction of correct labels.

The relationship between the F1- and ROC-AUC-optimized regimes is more subtle. In several model-dataset combinations, the two searches converge on the exact same threshold: Z-Score reproduces identical confusion matrices under both regimes on Datasets 1, 3, and 4, and Trigram NO POS does so on Dataset 1. Where the two regimes agree, F1 is evidently not being distorted by class prevalence, and either objective can be used interchangeably. Where they disagree, however, the disagreement is diagnostic: Z-Score on Dataset 2 is the starkest example, with F1-optimization settling on a threshold that flags nearly every instance as offensive (G-Mean 0.23) while ROC-AUC-optimization finds a materially more balanced alternative (G-Mean 0.73) at only a modest Accuracy cost (83.99% vs. 75.35%). The same disparity is observed on Datasets 2 and 4, but more markedly, in the case of Trigram NO POS. In each such instance the divergence is on a set where the class is, in one extreme case, the majority, or, in one of the cases cited (2), the label which a model is most likely to default to, and which is precisely the label to which it would be most likely to fall if there were no discrimination at all except the prevalence of the label. In all three regimes, NO Trigram and Trigram are the only two approaches that, for both objectives, have a Balanced Accuracy and G-Mean within a narrow band (typically less than 0.10), the most compelling evidence that their benefit is due to the genuine class separability of the underlying scores, rather than a favourable operating point chosen by the evaluator. The metrics for Pentagram and Trigram NO POS, on the other hand, are the least stable across the regimes, and even the best thresholds under F1- or ROC-AUC-optimization do not close the gap to NO Trigram and Trigram, further strengthening the point that the main limitation of the higher-order is sparsity-driven score degradation, rather than threshold selection.

3.7.Overall ROC AUC

Table 7. Overall ROC AUC (threshold-independent, computed on continuous scores)

Approach	1st Dataset	2nd Dataset	3rd Dataset	4th Dataset
Z-Score	0.75	0.78	0.63	0.73

Trigram NO POS	0.76	0.77	0.55	0.64
NO Trigram	0.95	0.94	0.72	0.84
Trigram	0.96	0.89	0.71	0.76
Pentagram	0.92	0.72	0.66	0.60

This threshold-independent ROC AUC, computed directly on continuous scores rather than under any single threshold regime, lets us summarize performance across the three threshold regimes above. Across the four datasets, the NO Trigram model is the strongest performer on three of them (Datasets 2–4, by both overall AUC and F1-optimized G-Mean), and a close second on the fourth. On Dataset 1, Trigram narrowly outperforms NO Trigram on every aggregate measure (AUC 0.96 vs. 0.95; F1-optimized G-Mean 0.89 vs. 0.89; F1-optimized Balanced Accuracy 0.89 vs. 0.89), so the addition of POS tags is not uniformly unhelpful, it is dataset-dependent, with NO Trigram the more reliable choice overall.

We had hypothesized that enriching trigram features with POS tags would improve contextual discrimination. With one exception (Dataset 1, discussed above), the Trigram model's performance showed no consistent improvement over NO Trigram, and on the highly skewed Dataset 4 it was clearly worse by every aggregate measure we report (F1 0.42 vs. 0.47; overall AUC 0.76 vs. 0.84; F1-optimized G-Mean 0.60 vs. 0.70), despite a marginally higher accuracy (91.72% vs. 90.85%) that is itself a symptom of the imbalance problem rather than a genuine improvement, since accuracy alone rewards correctly labeling additional majority-class examples. This outcome is consistent with the sparsity argument: adding POS tags effectively multiplies the feature space, since the same three-word sequence produces distinct features depending on grammatical role assignments. In sparse, imbalanced corpora, this fragmentation reduces co-occurrence counts per feature below reliable estimation thresholds, causing Laplace-smoothed probabilities to converge toward the prior rather than capturing genuine class-discriminative signal. Syntactic granularity, rather than sharpening discrimination, dilutes it under these conditions.

3.8.ROC Curves

The ROC curves in the Table 8 further validate the above results. The Z-Score Baseline (AUC 0.73) has some discriminating power for Dataset 4, and the curve is closest to the diagonal y = x (AUC 0.60) is near random discrimination. The NO Trigram variant gives the best curve shape, having a steep rise towards the top-left corner, which indicates high true positive rates at low false positive rates. The Trigram model is very similar and on three of the four datasets, the Trigram NO POS model is worse than the Pentagram model, though even this sparse model generates a more even curve. The validation of these ROC trends confirms two of our initial hypotheses: (1) simple lexical signals that can be extracted by the unigrams are indeed strong indicators of bias, and (2) the classification performance does not significantly improve by introducing syntactic granularity in the form of POS tags.

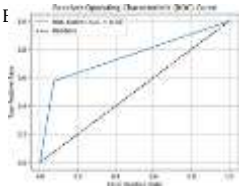
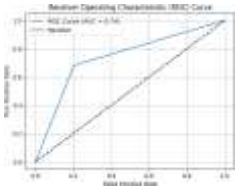
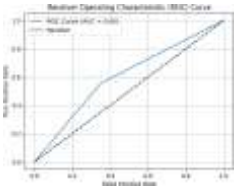
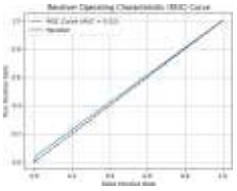
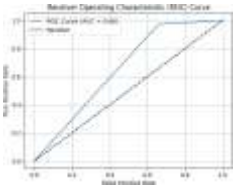
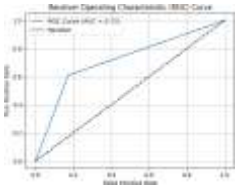
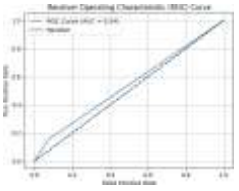
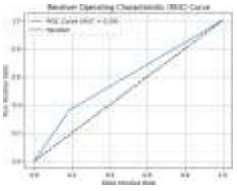
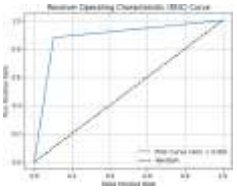
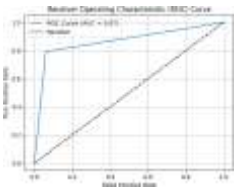
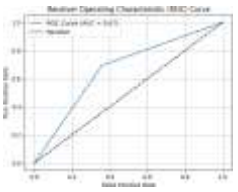
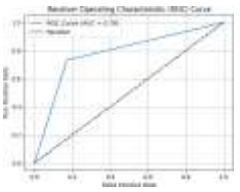
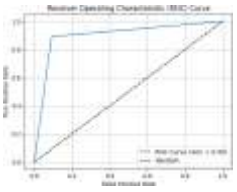
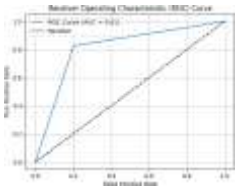
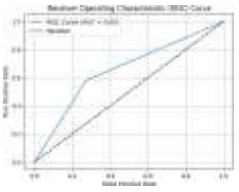
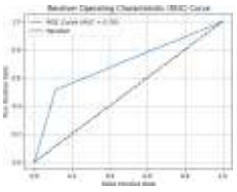
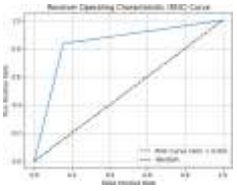
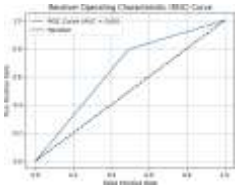
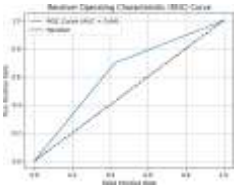
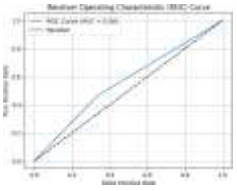
The curves also reveal a pattern in the data that is not directly apparent in the AUC table (Table 7): on Dataset 3, each approach's curve is clearly closer to the diagonal than on Datasets 1, 2, and 4, reflecting the fact that the AUC values for Dataset 3 are the lowest across all approaches (0.55–0.72, compared with 0.60–0.96 in other datasets). The smallest of the four corpora (5,000 posts) and the most lexically homogeneous, Dataset 3 is made up of highly formulaic political social media posts, which is likely to leave less distinctive vocabulary for all five approaches to exploit, irrespective of the n-gram order. This suggests that the sparsity effects discussed throughout this section are compounded, rather than caused outright, by corpus size and genre: even NO Trigram, the strongest approach overall, achieves its lowest AUC (0.72) on this dataset.

3.9.Interpretability of the Log-Likelihood Framework

A central motivation for adopting a log-likelihood framework over black-box classifiers is that every prediction can be decomposed into the individual feature scores that produced it (Section 1.1.). We demonstrate this concretely by ranking the trained feature set for each model by its log-likelihood score (Equation 6, stored in the bias score dictionary) and inspecting the extremes of that ranking: the highest-scoring features are those most strongly associated with the biased class, while the lowest-scoring features are most strongly associated with the non-biased class.

A direct pass over the trained feature set, however, surfaces a large number of features supported by only one or two training occurrences, and these low-count features can dominate both ends of the ranking for reasons that differ between the two scoring schemes. Under the Laplace-smoothed log-likelihood models, add-one smoothing prevents zero probabilities, but a feature observed only once or twice, especially one occurring exclusively in the minority class, can still receive a score of outsized magnitude relative to the limited evidence behind it. Restricting the ranking to features occurring at least τ times in the training set (we use $\tau = 5$) is sufficient to remove this artifact: under NO_Trigram, the resulting scores are well-bounded (Table 9), all falling within ±3.

Table 8. ROC Curves for Each Approach Across Datasets

Approach	1st Dataset	2nd Dataset	3rd Dataset	4th Dataset
Z-Score				
Trigram NO POS				
NO Trigram				
Trigram				
Pentagram				

The Z-Score Baseline exhibits the failure derived in Section 2.1. (Equation 1): count filtering alone does not resolve it, since the 1/P (w) term amplifies any relatively rare word by three to four orders of magnitude regardless of its occurrence count. This is borne out empirically: even after restricting to features with τ = 5 occurrences, the Z-Score Baseline's top-ranked features on Dataset 3 still take values in excess of 104 (e.g., branch, n=9 occurrences, score +16845.69), several orders of magnitude larger than the log-likelihood models' scores, with multiple features tied at identical extreme values, evidence of the same underlying instability rather than genuinely distinct signals. We therefore report the interpretability table only for the log-likelihood models, where scores remain well-calibrated after filtering.

Table 9 reports the resulting features for Dataset 3 (political social media posts, Section 3.) under the NO_Trigram configuration, the best-performing model overall; each entry can be traced directly back to the specific lexical feature driving the score, in contrast to the entangled, non-attributable predictions of a Transformer-based classifier. The top-ranked features (unilaterally, veto, threaten, king, notion) reflect language associated with executive authority and inter-branch conflict, consistent with Dataset 3's framing of partisan political discourse, while the bottom-ranked features (facility, tour, ceremony, Hope, serving) are dominated by routine constituent-service language, illustrating that the model's learned associations are semantically coherent rather than arbitrary.

Table 9. Top-10 and Bottom-10 Log-Likelihood Features, NO_Trigram on Dataset 3 (Political) (min. training count τ = 5)

Top 10 (most bias-indicative)			Bottom 10 (most non-bias-indicative)		
Rank	Feature	Score	Rank	Feature	Score
1	unilaterally	+2.7449	1	facility	−2.8609
2	fixed	+2.6114	2	tour	−2.7685
3	costly	+2.4572	3	Hope	−2.6304
4	king	+2.4572	4	ceremony	−2.5534
5	threaten	+2.4572	5	serving	−2.3791
6	notion	+2.4572	6	evening	−2.2790
7	veto	+2.4572	7	II	−2.2790
8	branch	+2.1695	8	Park	−2.2790
9	hurt	+2.1318	9	Dakota	−2.2249
10	ignore	+2.1318	10	Ebola	−2.2249

Table 10. Top-10 and Bottom-10 Log-Likelihood Features, Trigram NO POS on Dataset 3 (min. training count τ = 5)

Top 10 (most bias-indicative)			Bottom 10 (most non-bias-indicative)		
Rank	Feature	Score	Rank	Feature	Score
1	Congress work together	+1.3734	1	BEG I honored	−2.0766
2	health care END	+1.1221	2	public service END	−1.9224
3	broken immigration system	+0.9679	3	I hope join	−1.8046
4	health care cost	+0.8344	4	town hall meeting	−1.8046
5	BEG I bill	+0.7856	5	BEG I deeply	−1.6711
6	BEG I spoke	+0.5625	6	BEG I recently	−1.6711
7	BEG amp I	+0.5625	7	telephone town hall	−1.6711
8	BEG I stand	+0.5625	8	U.S. Capitol END	−1.6711
9	I sent letter	+0.5625	9	Dr. Martin Luther	−1.6711
10	equal pay equal	+0.4979	10	look forward seeing	−1.6711

Table 11. Top-10 and Bottom-10 Log-Likelihood Features, Trigram (POS) on Dataset 3 (min. training count τ = 5)

Top 10 (most bias-indicative)			Bottom 10 (most non-bias-indicative)		
Rank	Feature	Score	Rank	Feature	Score
1	DET law PUNCT	+3.0724	1	PROPN Service PROPN	−2.8625
2	ADP immigration NOUN	+3.0724	2	PRON in VERB	−2.6577
3	DET executive NOUN	+2.7359	3	ADJ news ADP	−2.6577
4	ADJ support ADP	+2.5128	4	DET very ADJ	−2.5111
5	DET IRS ADP	+2.5128	5	ADJ school NOUN	−2.4570
6	VERB law PRON	+2.3792	6	INTJ join PRON	−2.3999
7	PROPN need PART	+2.3051	7	BEG join PRON	−2.3999
8	PART a NOUN	+2.2251	8	PRON staff AUX	−2.3393
9	PART do AUX	+2.2251	9	PRON family CCONJ	−2.3393
10	AUX support DET	+2.2251	10	VERB I NOUN	−2.3393

The POS-augmented strategies tell a less favorable story. Trigram's top and bottom features (Table 11)

retain a recognizable lexical anchor in the middle slot (law, immigration, executive, support), but the surrounding POS tags contribute little beyond generic syntactic scaffolding (DET ... PUNCT, ADP ... NOUN).
The same procedure applied to the multi-word strategies (Tables 10–12) shows why the choice of feature representation matters for interpretability, not only for raw performance. Trigram NO POS (Table 10) surfaces genuinely readable, topically coherent phrases: broken immigration system, health care cost, and equal pay equal at the biased end, versus civic-courtesy phrases like public service, town hall meeting, and look forward seeing at the non-biased end. These are exactly the kind of contextual units, distinguishing a substantive policy phrase from a superficially similar unigram overlap, that motivated moving beyond unigrams in the first place; several entries also still carry the padding tokens BEG/END fused into the feature itself (e.g., BEG I honored, health care END), a direct illustration of the padding-bloat concern raised earlier in this paper (Section 2.).

Table 12. Top-10 and Bottom-10 Log-Likelihood Features, Pentagram on Dataset 3 (min. training count τ = 5)

Rank	Top 10 (most bias-indicative) Feature	Score	Rank	Bottom 10 (most non-bias-indicative) Feature	Score
1	PROPN PUNCT R PUNCT NUM	+2.6121	1	PROPN PROPN in PROPN ADP	−2.6862
2	PRON AUX concerned ADP DET	+2.1013	2	PROPN PROPN be DET ADJ	−2.5809
3	PROPN PUNCT D PUNCT PROPN	+1.7759	3	DET NOUN at PROPN PROPN	−2.5237
4	NOUN ADP a NOUN PRON	+1.5135	4	BEG VERB a NOUN ADP	−2.4631
5	NOUN PRON need PART AUX	+1.4081	5	NOUN NOUN at PROPN PROPN	−2.3985
6	NOUN AUX not DET NOUN	+1.4081	6	PROPN NOUN at PROPN PROPN	−2.3295
7	DET NOUN be DET NOUN	+1.4081	7	BEG PRON have DET ADJ	−2.3295
8	PRON AUX not VERB ADV	+1.4081	8	BEG BEG congratulation ADP PROPN	−2.2554
9	VERB DET law ADP DET	+1.4081	9	ADP PRON service ADP PRON	−2.2554
10	PART VERB a NOUN NOUN	+1.4081	10	PROPN PROPN in PROPN PUNCT	−2.2554

By Pentagram (Table 12), this pattern has largely taken over: several of the top and bottom features contain little to no lexical content at all, e.g., PROPN PROPN in PROPN ADP or DET NOUN be DET NOUN, and one bottom feature, BEG BEG congratulation ADP PROPN, again shows padding tokens directly forming part of a "top" feature rather than being incidental to it. This progression from lexically rich (Trigram NO POS) to syntactically dominated (Pentagram) features is a direct, feature-level illustration of the sparsity argument made in Section 2.1. and revisited throughout the Results: as window size and tag granularity increase, an increasing share of the model's highest-confidence evidence is drawn from syntactic patterning rather than the underlying words, which is consistent with these strategies' weaker aggregate performance (Tables 4–7).
In sum, these results reveal that fixed-window n-grams, whether simple or extended, struggle to capture nuanced bias patterns under class imbalance. Context beyond unigrams must be modeled in a way that avoids sparsity, and richer non-n-gram representations are needed to capture the subtleties of bias under class imbalance.

4.CONCLUSION

This paper empirically shows that adding more n-gram complexity does not necessarily lead to better class-bias detection in the presence of class imbalance. Across the 4 datasets with varying skew, the unigram model consistently performed the best overall in terms of precision and recall, whereas the trigram and pentagram variants introduced feature sparsity, leading to a drop in performance for minority classes, particularly in Dataset 4 and 8% toxic, with the Pentagram variant retaining over 92% accuracy but only an F1 score of 0.18. There was no measurable difference between the use of POS-augmented and plain trigrams, indicating that syntactic labelling is not enough to boost discrimination under sparsity. These results provide an empirically based starting point and directly inspire the development of more promising approaches to modelling higher-level contextual cues that n-grams are not good at capturing in

realistic, imbalanced social media settings, such as topic-aware and semantic context models including LDA, NMF, NER-based categorization, and topic assignment using LLMs.

Furthermore, our results show empirically that one inherent drawback of fixed-window n-grams is that they cause feature sparsity which negatively affects the results of minority classes in the case of class imbalance, as observed in all four datasets above. In addition, consistent with the general concerns expressed by Hovy et al. [19], the same sparsity mechanism likely extends to cross-lingual settings, since fixed-window n-grams would be expected to amplify noise in morphologically rich languages (e.g., Hindi) and over-represent patterns in resource-rich languages (e.g., English); we did not explore any non-English or multilingual dataset in this study, so this extension is an educated guess based on our findings, and is explicitly stated as a suggestion for further empirical research. Despite the fact that n-grams are good at learning local lexical items, they are not suitable for learning general lexical items.

Without careful attention to cues, they do not transfer subtle patterns of bias without compromising the accuracy of minority class performance. This paper was able to set the groundwork for better baseline performance for bias detection, but also showed that traditional feature-based approaches to context integration are not sufficient. Future work should focus on developing more advanced approaches to preserve interpretability while exploring new approaches to capture context, as well as a controlled comparison of resource cost with Transformer-based methods. The latter direction is topic-aware, category-based bias modeling, which is a promising one. This is not only based on surface n-gram features, but also on a coarse-grained thematic context as an intermediate layer. For instance, each document can be assigned to one or more of the high-level topics (e.g., “immigration,” “gender,” “hate ideology”) and a categorical bias score can be inferred based on how often these themes appear and how they are combined.

Topic categories can be derived through various interpretable or semi-structured techniques, including:

- **Latent Dirichlet Allocation (LDA):** Unsupervised topic discovery using word co-occurrence patterns.
- **Non-negative Matrix Factorization (NMF):** Factor-based dimensionality reduction with interpretable topic vectors.
- **Named Entity Recognition (NER):** Extraction of entity types (e.g., religion, nationality, gender) as proxies for bias dimensions.
- **Knowledge Graphs (e.g., DBpedia, ConceptNet):** Mapping entity mentions to structured knowledge bases for contextual grouping.
- **LLM-Based Classification:** Using large language models to perform zero-shot or few-shot categorization of comments into thematic or bias-relevant topics offers powerful new capabilities for bias detection. However, the deployment of such models also introduces significant risks and challenges related to fairness, trustworthiness, and unintended harms, as highlighted in recent work [30–32]. Thus, any adoption of such models should be accompanied by rigorous evaluation and mitigation strategies for bias, leveraging their powerful capabilities.

By empirically characterizing the limits of n-gram-based bias detection under class imbalance, this work establishes a foundation for developing future bias detection systems that balance contextual awareness and interpretability, while making explicit where fixed-window n-gram representations lose robustness in real-world social media settings.

References

1. S. Barocas, M. Hardt, and A. Narayanan, Fairness and Machine Learning: Limitations and Opportunities. MIT Press, 2023.
2. N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, “A survey on bias and fairness in machine learning,” 2022. [Online]. Available: https://arxiv.org/abs/1908.09635
3. L. Slechten, C. Courtois, L. Coenen, and B. Zaman, “Adapting the selective exposure perspective to algorithmically governed platforms: The case of google search,” Communication research, vol. 49, no. 8, pp. 1039–1065, 2022.
4. T. Sun, A. Gaut, S. Tang, Y. Huang, M. ElSherief, J. Zhao, D. Mirza, E. Belding, K.-W. Chang, and W. Y. Wang, “Mitigating gender bias in natural language processing: Literature review,” arXiv preprint arXiv:1906.08976, 2019.
5. A. Chouldechova and A. Roth, “The frontiers of fairness in machine learning,” arXiv preprint arXiv:1810.08810, 2018.
6. N. Sambasivan, E. Arnesen, B. Hutchinson, T. Doshi, and V. Prabhakaran, “Re-imagining algorithmic fairness in india and beyond,” in Proceedings of the 2021 ACM conference on fairness, accountability, and transparency, 2021, pp. 315–328.
7. V. Richter, C. Katzenbach, and J. Zeng, “Negotiating ai(s) futures: Competing imaginaries of ai by stakeholders in the us, china, and germany,” JCOM, vol. 24, no. 2, p. A08, 2025. [Online]. Available: https://doi.org/10.22323/2.2402020
8. P. Tsimpoukis, “Contesting dominant ai narratives on an industry-shaped ground: Public

discourse and actors around ai in the french press and social media (2012–2022)," JCOM, vol. 24, no. 2, p. A10, 2025. [Online]. Available: https://doi.org/10.22323/2.24020210
9. P. G. Rao, H. Chigurupati, K. Hashia, J. Thriveni, P. Deepa Shenoy, and K. R. Venugopal, "Detecting bias in social media using sentiwordnet," in Data Management, Analytics and Innovation, S. Goswami, S. Saha, K. Basu, and R. S. Beed, Eds. Singapore: Springer Nature Singapore, 2026, pp. 197–209.
10. H. Ahmed, I. Traore, and S. Saad, "Detection of online fake news using n-gram analysis and machine learning techniques," in Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments: First International Conference, ISDDC 2017, Vancouver, BC, Canada, October 26-28, 2017, Proceedings 1. Springer, 2017, pp. 127–138.
11. E. Ghadery, S. Movahedi, H. Faili, and A. Shakery, "MNCN: A multilingual ngram-based convolutional network for aspect category detection in online reviews," in Proceedings of the AAAI conference on artificial intelligence, vol. 33, no. 01, 2019, pp. 6441–6448.
12. J. Mutinda, W. Mwangi, and G. Okeyo, "Lexicon-pointed hybrid n-gram features extraction model (lenfem) for sentence level sentiment analysis," Engineering Reports, vol. 3, no. 8, p. e12374, 2021.
13. A. Ridwan, H. H. Nuha, and R. Dharayani, "Sentiment analysis of floods on twitter social media using the naive bayes classifier method with the n-gram feature," in 2022 International Conference on Data Science and Its Applications (ICoDSA). IEEE, 2022, pp. 114–118.
14. D.-H. Vu and T. Le, "Enhanced framework for detecting Vietnamese hate and offensive spans," IAES International Journal of Artificial Intelligence, vol. 15, no. 1, pp. 962–971, 2026. [Online]. Available: https://doi.org/10.11591/ijai.v15.i1.pp962-971
15. W. M. S. Yafooz, A. E. Yahya, A. Alsaeedi, R. Alluhaibi, F. Jamil, and M. S. Elsayed, "Exploring social media sentiment patterns for improved cyberbullying detection," IAES International Journal of Artificial Intelligence, vol. 14, no. 5, pp. 4211–4225, 2025. [Online]. Available: https://doi.org/10.11591/ijai.v14.i5.pp4211-4225
16. R. B. Harish and N. Siddalingaiah, "Semantic-syntactic graph network for aspect-based sentiment analysis," IAES International Journal of Artificial Intelligence, vol. 15, no. 2, pp. 1814–1824, 2026. [Online]. Available: https://doi.org/10.11591/ijai.v15.i2.pp1814-1824
17. M. Ziaulla and A. Biradar, "A hybrid model for enhanced aspect-based sentiment analysis using large language models," IAES International Journal of Artificial Intelligence, vol. 15, no. 2, pp. 1825–1838, 2026. [Online]. Available: https://doi.org/10.11591/ijai.v15.i2.pp1825-1838
18. E. Ferrara, "Fairness and bias in artificial intelligence: A brief survey of sources, impacts, and mitigation strategies," Sci, vol. 6, no. 1, p. 3, 2023.
19. D. Hovy and S. Prabhumoye, "Five sources of bias in natural language processing," Language and linguistics compass, vol. 15, no. 8, p. e12432, 2021.
20. P. G. Rao, H. Chigurupati, K. Hashia, T. J, P. D. Shenoy, and V. K. R., "Detecting bias in social media using SentiWordNet," Journal of Computer Science, vol. 22, no. 5, pp. 1552–1568, 2026.
21. S. Bird, "Nltk: the natural language toolkit," in Proceedings of the COLING/ACL 2006 interactive presentation sessions, 2006, pp. 69–72.
22. G. A. Miller, "Wordnet: a lexical database for english," Communications of the ACM, vol. 38, no. 11, pp. 39–41, 1995.
23. Y. Vasiliev, Natural language processing with Python and spaCy: A practical introduction. No Starch Press, 2020.
24. T. Mandl, S. Modha, P. Majumder, D. Patel, M. Dave, C. Mandlia, and A. Patel, "Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in indo-european languages," in Proceedings of the 11th Annual Meeting of the Forum for Information Retrieval Evaluation, ser. FIRE '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 14–17. [Online]. Available: https://doi.org/10.1145/3368567.3368584
25. M. Dalvi, "Twitter sentiments analysis (nlp)," https://www.kaggle.com/datasets/mayurdalvi/twitter-sentiments-analysis-nlp, 2024, accessed: 2024-10-05, Licensed under CC0.
26. A. Founta, C. Djouvas, D. Chatzakou, I. Leontiadis, J. Blackburn, G. Stringhini, A. Vakali, M. Sirivianos, and N. Kourtellis, "Large scale crowdsourcing and characterization of twitter abusive behavior," CoRR, vol. abs/1802.00393, 2018. [Online]. Available: http://arxiv.org/abs/1802.00393
27. T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated hate speech detection and the problem of offensive language," in Proceedings of the 11th International AAAI Conference on Web and Social Media (ICWSM). AAAI Press, 2017, pp. 512–515. [Online]. Available: https://github.com/t-davidson/hate-speech-and-offensive-language
28. CrowdFlower, "Political social media posts," https://www.kaggle.com/datasets/crowdflower/political-social-media-posts, 2016, ac-cessed:

2024-05-30.
29. cjadams, D. Borkan, inversion, J. Sorensen, L. Dixon, L. Vasserman, and nithum, "Jigsaw unintended bias in toxicity classification," https://kaggle.com/competitions/jigsaw-unintended-bias-in-toxicity-classification, 2019, kaggle.
30. R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, E. Brynjolfsson, S. Buch, D. Card, R. Castellon, N. Chatterji, A. Chen, K. Creel, J. Q. Davis, D. Demszky, C. Donahue, M. Doumbouya, E. Durmus, S. Ermon, J. Etchemendy, K. Ethayarajh, L. Fei-Fei, C. Finn, T. Gale, L. Gillespie, K. Goel, N. Goodman, S. Grossman, N. Guha, T. Hashimoto, P. Henderson, J. Hewitt, D. E. Ho, J. Hong, K. Hsu, J. Huang, T. Icard, S. Jain, D. Jurafsky, P. Kalluri, S. Karamcheti, G. Keeling, F. Khani, O. Khattab, P. W. Koh, M. Krass, R. Krishna, R. Kuditipudi, A. Kumar, F. Ladhak, M. Lee, T. Lee, J. Leskovec, I. Levent, X. L. Li, X. Li, T. Ma, A. Malik, C. D. Manning, S. Mirchandani, E. Mitchell, Z. Munyikwa, S. Nair, A. Narayan, D. Narayanan, B. Newman, A. Nie, J. C. Niebles, H. Nilforoshan, J. Nyarko, G. Ogut, L. Orr, I. Papadimitriou, J. S. Park, C. Piech, E. Portelance, C. Potts, A. Raghunathan, R. Reich, H. Ren, F. Rong, Y. Roohani, C. Ruiz, J. Ryan, C. Ré, D. Sadigh, S. Sagawa, K. Santhanam, A. Shih, K. Srinivasan, A. Tamkin, R. Taori, A. W. Thomas, F. Tramèr, R. E. Wang, W. Wang, B. Wu, J. Wu, Y. Wu, S. M. Xie, M. Yasunaga, J. You, M. Zaharia, M. Zhang, T. Zhang, X. Zhang, Y. Zhang, L. Zheng, K. Zhou, and P. Liang, "On the opportunities and risks of foundation models," 2022. [Online]. Available https://arxiv.org/abs/2108.07258
31. Y. Liu, Y. Yao, J.-F. Ton, X. Zhang, R. Guo, H. Cheng, Y. Klochkov, M. F. Taufiq, and H. Li, "Trustworthy LLMs: a survey and guideline for evaluating large language models' alignment," arXiv preprint arXiv:2308.05374, 2023.
32. E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell, "On the dangers of stochastic parrots: Can language models be too big?" in Proceedings of the 2021 ACM conference on fairness, accountability, and transparency, 2021, pp. 610–623.