OST
Eastern Switzerland
University of Applied Sciences

Hate Speech Detection with LLM

Bias Mitigation Using an Ensemble, In-Context Learning and Fine-Tuning

Kailing Peng, Lukas Derungs

Ost – Eastern Switzerland university of Applied Science

Advisor: Prof. Dr. Daniel Patrick Politze

December 20, 2024

OST
Eastern Switzerland
University of Applied Sciences

Table of Content

1. Abstract

The increasing acceptance of hate expression, particularly through the internet and social media, has significantly amplified the presence of online hate speech. This phenomenon negatively impacts psychological health and can incite violence, necessitating effective detection methods to prevent such content from being posted or shared. A recent trend involves leveraging the large context windows of new Large Language Models (LLMs) to enable In-Context Learning (ICL) where correct examples showcased in the prompt window. Studies have demonstrated the ability of LLMs in hate speech detection using few-shot strategies, comparing their performance to fine-tuned models. However, bias is a prevalent issue in pre-trained LLMs, requiring identification and mitigation specific to the task or dataset.

In this study, an experiment is conducted on text-based hate speech detection using ICL for two LLMs (GPT-4.o mini and Llama 3.1 8B) alongside a fine-tuned BERT model, forming a voting ensemble. The objective of this study is to assess whether bagging with majority voting can balance the strengths and weaknesses of individual models, thereby mitigating specific biases and achieving fairer, more accurate categorization of hate speech, offensive language and normal language.

The study results demonstrate that, while all LLMs exhibit different biases toward various labels (e.g., normal language being marked as offensive), majority voting effectively reduces bias and improves accuracy in categories where the participating models were close in performance. However, in scenarios where the performance of the participating models varies drastically, for example in the category offensive in this study, majority voting does not outperform the best single model. Furthermore, the voting ensemble encountered cases where a draw occurred, comprising about 5.4% of the total 13’229 data entries. Separate considerations were made for these ambiguous cases - either excluding them from the evaluation or replacing them with the best model’s decision. Excluding ambiguous data entries helped improve accuracy and reduce bias in misclassification but did not fully achieve the task's purpose of reaching a comprehensive result. This study underscores the potential of ensemble approaches in enhancing the fairness and accuracy of hate speech detection systems.

2. Lay Summary:

The increasing acceptance of hate expression, particularly through the internet and social media, has significantly amplified the presence of online hate speech. This phenomenon can negatively impact ppsychological health and incite violence, necessitating effective detection methods to prevent such content from being posted or shared. The methods of doing so have changed drastically over the last few decades, going from human moderation, where every piece of content had to be examined by a human to automated moderation where a program automatically flags texts that, depending on the implementation, contain certain words or an overly hateful or offensive message. This study investigates whether combining multiple advanced artificial intelligence tools, known as Large Language models (LLMs), can improve the detection of such inappropriate content compared to relying on just one model.

In order to achieve this three LLMs were used to check if a given text sample should be considered “Hate Speech”, “Offensive Language” or “Normal Speech”. The three models chosen for this task are “GPT4.o mini”, “Llama 3.1 8B” and “BERT”. Each of these models has specific strengths and weaknesses which leads to them making different kinds of mistakes, which are referred to as “Bias” in this study. To help them make their decision, they are provide with definitions for those three categories and with correctly classified examples. After each model classified the text, a voting process determines the final classification. The most popular choice among the three models is selected as the final answer. The results from this voting system are then compared to the individual performance of each model to see which approach is more accurate and has less Bias. In this process it is possible for all 3 LLMs to vote for a different classification in which case a tie occurs. This study used two approaches for dealing with this situation:
The first excludes the text samples that resulted in a tie from the result.
The second uses the vote of the LLM that is know to have the highest accuracy as a tie-breaker, using that LLMs vote as the final answer in case of a tie.

The results of this study suggest, that such a voting-based approach can successfully improve the performance, both in accuracy and bias, in cases where the participating LLMs are close in individual performance. However, when one model was significantly better than the others, the voting system struggled to outperform the strongest individual model.

3. Introduction

Expressing hate has become socially acceptable. The widespread use of the internet and social media has increasingly brought hate speech online. A press release from December 2023 by the Federal Statistical Office (FSO) [1] shows that in Switzerland 60% of young people aged 15 to 29 reported being confronted with online hate speech in the last three months.

Hate speech impacts our psychological health and incites violence. A study conducted by Soral et al. [2] found that exposure to online hate speech led to increased aggressive tendencies and desensitization to violence among the audience. Many other studies have investigated and confirmed the link between online hate speech and hate crime. They confirmed a temporal correlation between generic hate speech and hate crimes motivated by various prejudices. This highlights the influence of virtual interactions, which are interconnected with human behaviour in offline society. Several papers [3] [4] indicate that these hateful virtual interactions may serve as early indicators of an increase in hate crimes.

3.1 Hate Speech Definition and Transmission

The three labels—hate speech, offensive and normal/neither—are commonly used in hate speech detection to address the varying degrees of harmful content in textual data. Recognizing the importance of distinguishing between these categories improves model performance in classifying nuanced harmful language is critical.

3.1.1 Hate Speech

While hate speech lacks a universally accepted definition, studies in the field [5] [6] [7] commonly define hate speech as having the following trait:

- incites violence or hate-directed towards a specific group of people-motivated by aspects of the group's identity (like race, religion, or gender)

Based on these traits, this study defines hate speech as:

Speech that contains threats, hatred or promotes violence against an individual or group based on their identity.

3.1.2 Offensive Language

Similarly, offensive language is highly context and culture dependent, which is mentioned in studies [7, P.1] [8]. Many studies handling a similar topic avoid providing a definition for the term "Offensive".

However, since this study uses multiple different models, the term has been specified to ensure a consistent framework for classifying data and to clearly distinguish offensive language from hate speech. For this purpose, traits commonly used in the definition of offensive language were collected across various web dictionaries. For example, Cambridge Dictionary [9] defines offensive as "causing someone to feel upset and angry, often because of being rude", while Collins Dictionary [10] describes something offensive as being "rude or insulting". Law Insider [11], which leans more into the legal aspect, provides a list of adjectives that fall under offensive: "blasphemous, obscene, indecent, insulting, hurtful, disgusting, morally repugnant".

In line with these definitions, the following definition for offensive language is used:

Speech that insults an individual using slurs, vulgar terms or abusive, discriminatory remarks but doesn`t threaten the victim or promotes violence against them.

3.1.3 Normal Speech

Content that is not covered by the definitions of "Hate Speech " and "Offensive Language".

3.2 Online Hate Speech Transmission and Platform Responsibility

3.2.1 Online Hate Speech Transmission

Hate speech may commonly be transmitted through social media, messaging platforms, forums, comment sections and visual content.

Platforms like Twitter, Facebook and Instagram allow users to share text, images and videos globally and instantaneously. The anonymity they offer, coupled with their broad reach, makes these platforms prime channels for hate speech. Research conducted by Mathew et al. [12] indicates that hateful content spreads more rapidly, widely and extensively than non-hateful content on social media, with hateful users demonstrating a higher degree of influence.

Messaging apps such as WhatsApp, Telegram and Signal provide encrypted communication, enabling the private sharing of hateful content within groups or directly between users. The study from Matos et al. [13] confirms that hate speech is both prevalent and problematic on these platforms, emphasizing the difficulty of regulating harmful discourse in encrypted environments.

Forums like Reddit, 4chan and various news website comment sections foster anonymity, often resulting in unchecked hate speech. The study by Rieger et al. [14] examined the extent, nature and clustering of hate speech in political fringe communities on Reddit, 4chan and 8chan finding that 24% of comments contained explicit or implicit hate speech.

Internet memes, GIFs and videos are particularly effective at subtly spreading hate speech, crossing language barriers. The ease with which visual content can be shared enables the rapid dissemination of harmful messages. The study by Aranda Serna [15] highlights the issue of hate speech propagation via memes on social networks, especially in political contexts where memes have played a significant role in shaping public opinion, such as during election campaigns.

3.2.2 Platform Accountability and Regulatory Measures

The responsibility of tech companies to combat online hate speech is no longer merely a matter of social agreement but is now increasingly shaped by legal expectations. Calls for stronger regulations and accountability urge platforms to consistently and transparently enforce their community guidelines.

In 2016, the European Commission together with Facebook, Twitter, YouTube and Microsoft signed a Code of Conduct to tackle the spread of illegal hate speech online in Europe [16]. As part of this voluntary agreement, these companies “committed to refining internal procedures and staff training, ensuring that the majority of valid reports for illegal hate speech are reviewed within 24 hours and necessary actions, such as removal or disabling access, are taken”.

In Germany, the Network Enforcement Act (NetzDG) [17], effective since 2018, mandates that social media platforms remove "obviously illegal" hate speech and harmful content within 24 hours of notification, or face significant fines. Platforms must also store deleted content for at least 10 weeks and provide transparency reports on how they handle illegal content every six months. Similarly, the UK’s Online Safety Act [18], passed on 26 October 2023, imposes new obligations on social media companies and search engines, with "abusive or hateful content" as a priority area for regulation. This law enhances platform accountability by holding companies more responsible for the safety of their users.

3.3 Major Challenges in Detecting Online Hate Speech

The detection of online hate speech faces numerous challenges across legal, business and technical domains, as outlined in the following sections.

3.3.1 Ambiguity, Context Dependence and Sarcasm

Hate speech can be ambiguous and context-dependent. The same word or phrase may be offensive in one context but innocuous in another. Detecting these nuances requires sophisticated models that understand the context. A study by Kumaresan et al. [19] highlighted the challenge of deconstructing ambiguity, “particularly when there are fewer overt hateful keywords present”. Additionally, detecting sarcasm and irony is particularly challenging for AI, as these often rely on subtle cues that are difficult to model accurately. In multilingual contexts, cultural nuances and language differences further complicate the interpretation of meaning.

3.3.2 Multilingual and Multicultural Challenges

Global platforms require hate speech detection in multiple languages and dialects. Each language has its own nuances, slang and cultural references. Furthermore, different cultures may have varying thresholds for what constitutes hate speech, adding another layer of complexity to detection and moderation processes. A Simona Frenda’s study from 2018 [20] investigated “the role of creative linguistic devices, particularly sarcasm, in hate speech within a multilingual context”.

3.3.3 Evolving Language and Slang

The dynamic nature of language, particularly evident in slang and memes on the Internet, poses serious challenges to the adaptability of LLMs. Models need to be continuously updated to keep pace with these changes. Users often use euphemisms, abbreviations and coded language to evade detection, making it a significant challenge to identify and decode such language. The study “SLANG: New Concept Comprehension of Large Language Models” from Mei et al. [21] introduced a benchmark for evaluating how well LLMs understand new slang and evolving language and also proposed a method named “FOCUS” for improving LLM comprehension of these concepts without retraining the models.

3.3.4 Balancing Accuracy and Freedom of Speech

Overly aggressive hate speech detection can lead to false positives, where benign content is incorrectly flagged as hateful, stifling legitimate expression and leading to user dissatisfaction. Conversely, under-detection results in false negatives, where actual hate speech goes undetected, harming users and damaging the platform's reputation.

3.3.5 Scalability and Variability

Hate speech is not always confined to static text; it can also appear in images, GIFs, audio and video comments. Detecting hate speech in these varied formats poses a significant technical challenge. Additionally, real-time detection is crucial for many platforms, but achieving this at scale for LLMs based approaches is difficult due to the large volume of user-generated content.

3.3.6 User and Legal Implications

Implementing hate speech detection must respect user privacy. Overly intrusive methods can lead to backlash and legal issues. Different jurisdictions have varying laws regarding hate speech, requiring platforms to navigate these legal landscapes carefully to comply with local regulations while maintaining a consistent global policy. The study done by Kapelańska-Pręgowska et al. [22] analyzed “the development of human rights standards regarding freedom of expression and hate speech in Poland and Slovenia through a comparative analysis of Polish and Slovenian law and practice”.

3.3.7 Data Limitations and Reliability

Most publicly available datasets are in English, limiting the applicability of models to other languages. For multilingual tasks, datasets are often translated from English, which can overlook cultural and linguistic

nuances, reducing their quality. Additionally, the trustworthiness of many open-source datasets is unclear and inconsistent labeling or unrepresentative data may lead to biased models. These biases can result in unfair outcomes, disproportionately flagging certain groups' speech as hateful while missing others. Large, high-quality and diverse datasets are expensive to collect and rarely available to the public for research purposes.

3.4 Evolution and Limitation of Hate Speech Detection Methods

The methods for detecting hate speech have evolved significantly over time, transitioning from manual moderation, where users flagged offensive content for human review, to increasingly sophisticated automated systems. Early automated methods often relied on rule-based approaches, using predefined keywords. These systems, while helpful for detecting explicit hate speech, struggled to understand context, leading to numerous false positives and negatives.

Chapter 4 of the Book “Social Media and Democracy” published in 2020 [23] provides a comprehensive overview of various automated approaches to hate speech detection. Combining the major challenges listed in the last chapter, the points discussed in the book are quoted and summarized below:

1. **Dictionary-Based Methods**: These methods rely on predefined lists of offensive terms, slurs, insults and their variations to identify hate speech. They count occurrences of these terms and may use distance metrics to detect misspellings or code words meant to avoid detection. However, dictionary-based methods are limited in detecting subtler or indirect forms of hate speech, such as sarcasm or evolving expressions. They also struggle with contextual nuances, making them less effective in recognizing more complex hate speech.
2. **Supervised Text Classification**: Techniques like Naive Bayes, Support Vector Machines (SVM), decision trees and random forests classify text based on features like word frequencies and n-grams (e.g., bigrams or trigrams). These methods perform well when trained on labelled data but can fail to address linguistic complexities like irony, sarcasm and coded language. They are also often platform-dependent, as different online platforms use distinct language styles, which can impact the model's ability to generalize across contexts.
3. **Topic Modelling and Sentiment Analysis**: These methods focus on identifying hate speech by analysing the topics being discussed (e.g., race, religion) and the tone of the content, typically negative or hostile. By detecting sentiment and associating it with specific topics, these methods can flag potentially harmful speech. However, they may struggle to detect hate speech without overt negative sentiment or in cases where the tone is more subtle.
4. **Word Embedding and Deep Learning**: Deep learning models such as doc2vec, FastText and neural networks can detect more complex patterns in text by understanding word relationships based on a deeper semantic analysis. While they can capture indirect hate speech, they require substantial labelled data for effective training and can be computationally intensive.
5. **Theoretically Motivated Approaches**: These approaches focus on identifying underlying ideological markers of hate speech, like "us vs. them" language and stereotypes. They provide valuable context for identifying certain forms of hate speech but may miss others that don't follow these ideological patterns.
6. **User Characteristics and Network Features**: This approach enhances hate speech detection by incorporating user data, such as the number of followers, friends and interactions within social networks. By analysing user behaviour and network dynamics, these methods can improve detection accuracy, as they consider the social context in which hate speech occurs. However, they may face privacy concerns and require significant amounts of data to be effective.

7. **Large Pre-Classified Datasets**: This technique uses vast amounts of data from known communities where hate speech is prevalent to build predictive models. The bag-of-communities method identifies speech patterns specific to these communities, improving the accuracy of detection. However, this method is highly dependent on the quality and scope of the data and it may struggle to adapt to emerging hate speech forms outside of these pre-defined communities.

Some of the recent developments in hate speech detection have focused on leveraging transformer-based models such as BERT (Bidirectional Encoder Representations from Transformers), GPT (Generative Pre-training Transformer) and their variants. These models undergo large-scale pretraining on diverse datasets, enabling them to better understand nuanced language patterns and context compared to earlier methods. Fine-tuning these models on specific hate speech datasets has shown promising results in improving detection accuracy and reducing biases [6].

4. LLMs for Hate Speech Detection

While machine learning (ML) and LLMs offer significant improvements, they are not a perfect solution and continue to face limitations in areas such as bias, context comprehension and adaptability to evolving hate speech.

4.1 Bias Existing in Pre-Trained LLMs

LLMs can exhibit biases and these biases arise from several factors. Primarily, the training data plays a significant role. LLMs are trained on vast amounts of text data sourced from the internet, books, articles and other written materials. If this data contains biases, stereotypes or prejudiced language, the models can learn and replicate these biases. For instance, if a dataset has more positive associations with one gender, race or cultural group over others, the model may reflect those biases in its outputs.

Additionally, certain groups may be underrepresented in the training data. Texts written by or about marginalized communities might be less prevalent, leading to models that perform poorly when dealing with topics related to these communities, further perpetuating bias. Algorithmic bias also plays a role; the processes of selecting, cleaning and curating training data often involve human judgment, which can be subjective and inadvertently introduce biases. Furthermore, the objective functions of LLMs, typically optimized for predicting the next word in a sentence or generating coherent text, do not account for fairness or bias mitigation, leading models to prioritize coherence and fluency over fairness and equity.

Historical and societal biases are also reflected in LLMs. Language reflects the values and norms of the society that uses it and LLMs trained on historical texts or content from biased sources will inherit these societal biases. Once deployed, LLMs can be influenced by the way they are used; if users interact with a model in a biased manner or if biased outputs are more frequently upvoted or liked, the model can reinforce and amplify these biases.

Examples of biases in LLMs include gender bias, such as associating certain professions predominantly with one gender (e.g., assuming a doctor is male and a nurse is female), racial bias, such as providing different quality or tone of responses based on racial cues, cultural bias, favouring certain cultural norms and values over others and political bias, reflecting the political leanings of the sources it was trained on.

Efforts to mitigate biases in LLMs include ensuring diverse and representative training datasets, developing techniques to identify and reduce biases in model outputs, involving human reviewers to correct biased outputs and increasing transparency around how models are trained and evaluated. Despite these efforts, completely eliminating bias from LLMs remains a challenging and ongoing area of research.

4.2 Methods Recommended to reduce bias in LLMs

There are multiple ways bias can be reduced, here is a non-exhaustive list:

4.2.1 Diverse Training Data

Ensuring training datasets are diverse and representative of different groups and perspectives is essential. However, this is often achievable only by the development company, as they control the data collection and curation processes.

4.2.2 Bias Detection and Mitigation Techniques

Developing techniques to identify and reduce biases in model outputs is a common approach. Methods such as adversarial training, fairness constraints and post-processing corrections are typically adopted. However, these methods often focus on specific types of bias reduction in specific datasets.

4.2.3 Human-in-the-Loop

Involving human reviewers to identify and correct biased outputs is another strategy. Annotators follow specific guidelines to make decisions or judgments and human-labelled answers in a dataset are usually used as ground truth for further investigation into LLM performance. While effective, this approach is very expensive.

4.2.4 Transparency and Accountability

Increasing transparency regarding how models are trained and evaluated and holding platforms accountable for biased outcomes, is crucial. Platforms such as Google AI have implemented transparency policies to address these concerns. Additionally, tools like attention visualization aim to illustrate how a model focuses on different parts of the input text when generating responses. This visualization allows us to identify which parts of the input most strongly influence the output, helping to uncover biases and better understand the model's decision-making patterns.

4.3 Bias in LLM for Hate Speech Detection

Bias is also evident in hate speech detection. Research has indicated that LLMs can exhibit excessive sensitivity to hateful content and show bias in labelling specific targets within certain datasets. For example, the study from M. Zhang et al. [29] highlights that LLMs may display heightened sensitivity towards certain groups or topics, which can lead to fairness issues by misclassifying benign statements as hate speech. Additionally, the same study points out calibration limitations, that LLMs' confidence scores often remain within a narrow range, regardless of the dataset’s complexity.

A study from Das et al. [24] highlights the presence of annotator biases in hate speech detection. Quoted from the study, “One potential direction involves mitigating these biases by incorporating specific rules into the LLMs while training or prompting annotators to prevent biased outputs. Additionally, exploring broader aspects of the problem statement through enhanced language style or lexical content analyses holds promise”. The study underscores the risk of biases emerging when LLMs are directly utilized for annotation tasks.

To understand the behaviour of LLMs, a test was conducted on GPT. The findings are as follows:

4.3.1 Fine-Tuning Possibility:

From GPT-2 onwards, no fine-tuned models specifically for hate speech detection were identified, likely due to content moderation policies introduced at a certain point. Attempts to bypass the content filter using

techniques like DAN (Do Anything Now) on version 3 or above were unsuccessful. Lower versions have less relevance for this study and were not tested. Uploading a dataset containing hate speech resulted in a job failure.

4.3.2 Moderation Standards and Sensitivity:

In the absence of specified guidance, GPT models rely on their own moderation standards seen in Figure 1. These standards identify potentially harmful content in text and images. Llama employs a similar moderation policy. However, the exact threshold standard for flagging content is not clearly stated.

To test the model's sensitivity, an experiment was conducted using 50 randomly selected data points, which included normal, offensive and hate speech instances with no examples or class definitions. The results indicated that, regardless of overall accuracy, GPT tended to misclassify normal instances as hate or offensive speech more frequently than the reverse (i.e., hate or offensive speech being misclassified as normal). Several normal instances misclassified as offensive or hate speech were selected and the moderation scores were generally high. It was noticed that base LLMs appear aggressive and Zero-shot classification exhibited noticeable label flipping frequency.


```
"id": "modr-970d409ef3bef3b70c73d8232df86e7d",
"model": "omni-moderation-latest",
"results": [
    "flagged": true,
    "categories": {
      "sexual": false,
      "hate": false,
      "self-harm": false,
      "self-harm/intent": false,
      ...}
    "category_scores": {
      "sexual": 2.34135824776394e-7,
      "hate": 3.1999824407395835e-7,
      "self-harm": 0.0011175734280627694,
      "self-harm/intent": 0.0006264858507989037,
      ...}
]
```


Figure 1 Simplified GPT Moderation

4.4 Project Direction

The study "Comparing Fine-Tuning, Zero and Few-Shot Strategies with Large Language Models in Hate Speech Detection in English" from Pan et al. [25] serves as the research inspiration for this study.

The same study investigates the ability of different LLMs to identify sexist and hate speech. It evaluates approaches ranging from zero-shot and few-shot learning to fine-tuning using two datasets: Explainable Detection of Online Sexism (EDOS) and Multilingual Detection of Hate Speech Against Immigrants and Women on Twitter (HatEval).

As part of the conclusion from the same study, “The error analysis in the study reveals that contextual learning struggles to distinguish between types of hate speech and figurative language. On the other hand, the fine-tuned approach often produces many false positives”.

4.4.1 Method and Hypothesis

To leverage the strengths of both fine-tuning and few-shot learning, this study proposes using an ensemble majority voting system with three LLMs: GPT-4.O mini (using in-context learning), Llama 3.1 8B (using in-context learning) and fine-tuned BERT. This technique is widely used in machine learning, enhances model robustness and reduces bias. This hypothesis of this study is that an ensemble voting result will be less biased, fairer and perform better in accurately labelling each category hate speech, offensive language, or normal language.

5. Bias Measurement

This project aims to address specific biases in the decision-making of LLMs when classifying labels in hate speech detection. However, this study does not delve into identifying the underlying factors that lead to biased judgments. The chosen approach focuses on comparing the LLM-generated labels with those in the original dataset, HateXplain [26].

5.1 Measured Bias

Bias against specific target groups or LLM annotator bias is not directly measured in this study. They have been extensively surveyed in existing literature. For instance, the paper from Garg et al. [27] explores bias based on both the source of the text and the target group. The study mentioned before [24] examines GPT-3.5, GPT-4o, Llama-3.1 and Gemma-2 to identify systematic tendencies or prejudices that LLM annotators exhibit when labelling data. For example, a gender bias is indicated if an LLM annotates a social media post as hate speech more frequently when it contains references to women compared to when it references men.

The primary focus of this study is to measure bias in the LLMs' decisions for label classification by comparing these decisions against the original labels in the HateXplain dataset. This approach allows the identification of deviations in the model's classifications from the annotated ground truth, providing insights into the biases present in the model's decision-making process.

5.2 Bias, Visual Representation

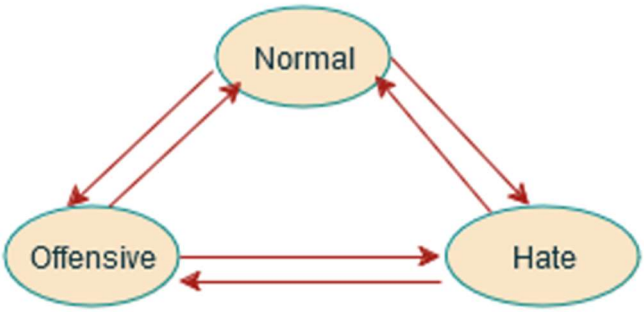


Figure 2 Bias Visual Representation

The arrows indicate the bias tendency of the model between categories.

Bias Example:

- Normal -> Offensive: The model tends to classify normal posts as offensive.
- Offensive -> Hate Speech: The model tends to classify offensive posts as hate speech.
- Hate Speech -> Normal: The model tends to classify hate speech as normal.

These arrow directions show where the model's bias lies in the task of categorizing the input posts.

6. Data

The HateXplain dataset is utilized, which is the first of its kind to combine hate speech labels with rationales, thereby enhancing the transparency and trustworthiness of hate speech detection models. Cited in over 500 papers to date according to google scholar [28], the dataset has proven to be a highly valuable resource for research in this field. Given that its labelling decisions are treated as the ground truth for this project, HateXplain serves as a reliable and authoritative source for training and evaluating models.

6.1 Dataset Details

The HateXplain dataset comprises 20’148 posts from social media platforms like Twitter and Gab, each labelled as “Hate Speech”, “Offensive”, or “Normal”. These posts were annotated by 253 crowd-workers, who provided not only the category label but also rationales - marked text spans that justify why a particular label was assigned. Additionally, the annotators identified the target community, specifying the group targeted by the hate speech if applicable (e.g., race, religion, gender).

A notable limitation of the HateXplain dataset is that each post was annotated by only three individuals, resulting in a relatively large step size of 33% when considering the voting percentages. Therefore, the dataset was categorized using majority voting, resulting in 5’935 posts labelled as Hateful, 5’480 as Offensive, 7’814 as Normal and 919 posts remaining undecided.

6.2 Dataset Preparation

The original HateXplain dataset was already very clean and well-prepared, requiring no data cleaning. After verifying its legitimacy, simple modifications were made to prepare it for this use case.

6.2.1 Label Aggregation and Simplification

Each data point in the HateXplain dataset contains a post_id, comprehensive voting details from all annotators and the post content represented in tokens. For the purposes of this project, the voting results are consolidated to retain only the majority voting outcome for each sentence. The tokens are concatenated into sentences and the complex, long post_ids are replaced with simple integer indices, starting from 1.

6.2.2 Exclusion of Undecided Instances

The undecided instances, where annotators did not reach a majority agreement, do not contribute to the performance measurement as the annotator-voted label serves as the ground truth. Consequently, these undecided instances are excluded from the dataset, ensuring that the ground truth is clear and unambiguous. This approach aligns with the methodology employed in the original HateXplain paper.

6.2.3 Division of the Dataset

The remaining 19’229 data points are divided into two datasets: a training set and a testing set. The training set comprises 6’000 posts, with an equal distribution of 2’000 posts from each category (Hate Speech, Offensive, Normal). This balanced representation helps to reduce bias and improve the model's generalization capabilities during BERT fine-tuning. The training set also serves as the example pool for the few-shot strategy in ICL.

The testing set, containing the remaining 13’229 posts, will be used to comprehensively evaluate the model's performance across all LLMs. Specifically, the testing set includes 3’935 posts labelled as Hateful, 3’480 as Offensive and 5’814 as Normal.

For GPT and Llama models, which adapt ICL instead of training, only a testing set is required for the evaluation. While for BERT fine-tuning, the training data is further split into 80% for training and 20% for

validation utilizing 5-fold cross-validation to ensure robustness and accuracy.

7. Experiment Design

The goal of the experiment design is to implement and validate strategies for guiding the responses of LLMs in a desired manner. This section outlines the methods and factors considered to ensure effective performance of the models in detecting and classifying hate speech.

7.1 Method Few-Shot ICT

In this approach, In-Context Learning (ICL) is used, where models like GPT and Llama leverage a large context window and rely on carefully crafted prompts for few-shot learning. The models are not fine-tuned but instead they generate predictions based on “learning” from provided examples within the context.

7.1.1 Prompt Engineering

Prompt engineering is a strategic approach to guiding LLM’s output, leveraging models' contextual processing capabilities to enable adaptive, task-specific performance through minimal example-based learning. Through precise input structuring, setting restrictions and providing a few examples, LLMs can efficiently adapt to specialized tasks without extensive retraining. The methodology is particularly effective in complex domains like semantic analysis and content moderation, allowing models to interpret nuanced linguistic contexts with greater accuracy.

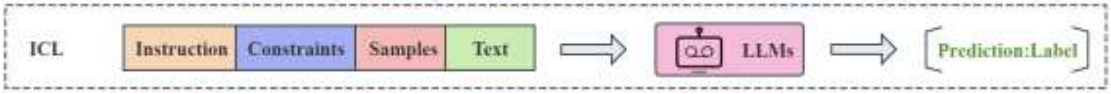


Figure 3 ICT Design

Figure 3 illustrates the components of an in-context learning (ICL) prompt. The draft system message in Figure 24 in Appendix B provides an example of the prompt used in this project.

7.1.2 Factors Affecting LLMs' Consistency and Reliability

The relevance and value of factors that affect an LLM's consistency and reliability need to be considered and analyzed for the implementation.

7.1.2.1 Temperature

LLM like Llama and GPT are autoregressive models that predict the next token in a sequence by generating logits, which represent token likelihoods. These logits are converted into a probability distribution using a SoftMax function depicted in Figure 4. The temperature parameter, ranging from 0 to 1 for GPT models and 0 to 2 for Llama models, controls the randomness of model responses by scaling the softmax output. The study from Zhang et al. [29] examined the effects of temperature on model outputs.

$$P(t_i) = \frac{e^{\text{logit}_i/T}}{\sum_j e^{\text{logit}_j/T}}$$

Figure 4 SoftMax Function.

Temperature Characteristics:

- Higher temperatures increase output diversity and creativity.
- Lower temperatures produce more conservative, deterministic results.
- Default temperature settings vary across different language models.

For this specific task, where consistent and reliable single-word outputs are required, temperature's impact requires careful consideration. Two primary concerns are:

1. Low temperature might potentially amplify inherent model biases, leading to skewed predictions.
2. Excessive randomness from a higher temperature might compromise the reliability and consistency of the output.

To identify the optimal temperature setting and understand its influence on model performance, testing will be conducted to evaluate its effects on output consistency and potential bias.

7.1.2.2 Top P (Nucleus Sampling)

As described in this article by Karl Weinmeister [36], top P or nucleus sampling, is a parameter that influences the randomness of LLM outputs, ranging from 0 to 1. It computes the cumulative probability distribution of tokens and truncates the pool when the cumulative probability exceeds the threshold. This method dynamically adjusts the set of possible tokens, concentrating on the most probable ones while still allowing for some diversity in the generated output. For instance, a Top P of 0.3 means only the tokens within the top 30% of the probability mass are considered. Since probability calculation is not applicable in this study, this factor is not relevant for consideration in the context of the task.

7.1.2.3 Context Window

The context window defines the number of tokens an LLM can process at once. A larger context window allows the model to retain more information from prior tokens, which is crucial for generating coherent and contextually accurate responses. Once the input exceeds the context window, earlier tokens are "forgotten," which can lead to irrelevant responses or reduced output quality.

For this experiment, data entries are grouped into patches, each consisting of a compact long-text array, which is processed via API calls that return an array of outputs (one output per input per API call). Given that each API call processes a single input and output, the model’s memory is not particularly relevant in this context. Instead, the focus is on the maximum token limit of the context window, which plays a more relevant role in ensuring output consistency and input/out size design.

7.1.3 Few-shot Learning

7.1.3.1 Rationale for 5-Shot Number

In the inspiration study of this experiment on hate speech binary classification, the 5-shot approach was found to yield the best results in both zero-shot and few-shot classifications when using prompts in generative models.

This result aligns with findings from study done by Guo et al. [30], which proposed four diverse prompting strategies to optimize the use of LLMs for context-aware hate speech detection. In their experiments, a balanced test dataset from HateXplain was created by randomly selecting 200 samples from each category

(Race, Religion, Gender, Sexual Orientation and Miscellaneous Hate Speech), alongside 1’000 samples labelled as ‘normal’, totalling 2’000 test samples. Samples labelled ‘offensive’ were excluded. The study applied an equal 5-shot approach for both hateful and non-hateful samples during prompt engineering and the results demonstrated that the 5-shot approach was highly effective. Furthermore, this study revealed that “increasing the number of shots beyond a certain threshold did not lead to significant improvements in performance”.

Based on these insights, the 5-shot approach was extended with five examples from each category for this study.

7.1.3.2 Rationale for Fixed/ Dynamic Examples:

In the inspiration study, it was observed “that using the first five examples of each label from the training set with the Zephyr model was highly effective for the EDOS set”. Additionally, “another finding from the EDOS set suggested that the 5-shot approach using random examples outperformed the 5-shot approach with the retrieval module, except in the case of detecting sexist texts”. However, the study did not offer a direct comparison between random versus fixed examples in the 5-shot approach, which limits the ability to extend these findings directly for the experiment. How this factor will affect the experiment is an aspect considered in the design testing phase.

Other Considerations:

The studies referenced above utilized a balanced dataset with the 5-shot approach. In contrast, the test set for this experiment is unbalanced, consisting of 3’935 hate speech samples, 3’480 offensive samples and 5,814 normal samples. The potential impact of this sample distribution on the effectiveness of prompt engineering remains unchecked. While this factor is considered, it will not be part of the evaluation in the scope of this project.

7.1.3.3 Rationale for Uniform Evaluation Across Models

Using a method that implements randomly built bootstrap datasets can lead to inconsistencies in decision-making for each data point. To ensure every data point receives a consistent classification, the entire dataset needs to be evaluated uniformly.

For example, when a data point is evaluated n times (even or odd), it will be selected randomly fewer than n times for evaluation. If the data is not selected at all, there will be no valid output. In cases of a tie, such as 3 votes for "Normal" and 3 votes for "Offensive," the outcome becomes undecided, which is also invalid. Uneven evaluation causes fairness issues, potentially leading to ties or conflicting decisions.Additionally, if a debatable data point is evaluated only once, the uncertain classification could significantly influence the result, raising concerns about fairness and trustworthiness.

Decision: To address these concerns, uniform evaluation is employed, where each model annotates every data point the same number of times over an odd number of iterations. This ensures that each data point is consistently evaluated by its model, leading to a fair evaluation and mitigating issues like ties or conflicting outcomes during label aggregation.

Given the possibility that a LLM might fail to answer a question, particularly when dealing with sensitive content, the potential for missing a vote or encountering a tie situation in aggregation cannot be eliminated, even with careful design planning. Achieving perfect label aggregation for each entry is not the primary focus of this study. Manual evaluation of every non-perfect aggregation will not be conducted as long as the aggregation leads to a decision. In tie situations, provided such cases occur no more than a handful of times per dataset, the first most voted option will be selected.

7.1.4 Batching Based on Maximum Tokens

LLMs have a maximum number of input tokens, which requires splitting the dataset into batches. These "batches" refer to local data splits used as the model input.

The GPT-4o mini supports a maximum context window of 128’000 tokens, with up to 16’384 output tokens per request. The Llama 3.1 8B model also supports a maximum context window of 128’000 tokens, which includes both input and output tokens. To stay within the model's limits, a batch size of 100 data points is chosen for a preliminary test, ensuring that the total number of input tokens remains below the maximum threshold, with sufficient capacity for output tokens.

7.1.5 Test On the effect of Temperature and Fixed/Random ICL Examples

To confirm the rationale, a test was conducted to examine the impact of temperature and the differences between fixed and random examples on classification consistency. The test was set up as follows:

- 100 Datapoints were chosen at random from the dataset.
- The temperature started at 0 and was increased by 0.1 after each iteration.
- Llama classified each of the 100 datapoints 20 times per temperature.

The following statistics were calculated:

- **accuracy =** $\frac{number\ of\ correct\ votes}{number\ of\ total\ votes}$

Accuracy measures how often the model correctly classified a data point across all iterations. This metric provides insight into the overall correctness of the model's predictions.

- **Highest Vote average =** $\frac{Number\ \text{of votes for the most popular class}}{total\ votes}$

This metric indicates how consistently the model classified the same text with the same label, irrespective of correctness, this is important as consistency is the main focus of the test. For example, if there were 15 votes for "normal", 5 for "offensive" and 0 for "hate speech", the highest vote average would be: $\frac{15}{20} = 75\%$

- **Correct votes average =** $\frac{Number\ \text{of votes for the correct class}}{total\ votes}$

This metric shows how often the model voted for the correct class, regardless of the final classification. For instance, if the true label of a data point is "offensive" and the vote distribution is: 15 votes for "normal", 5 for "offensive" and 0 for "hate speech", the correct votes average would be: $\frac{5}{20} = 25\%$. This differs from accuracy in that its calculated per datapoint instead of per dataset like in accuracy, where a single datapoint can only have a result of 0% or 100%.

- **Test Statistics from Llama 3.1 3B**

Temperature	Fixed			Random		
	Accuracy	Highest Vote average	correct votes average	Accuracy	Highest Vote average	correct votes average
0	57%	78.98%	55.46%	54%	76.99%	54.30%
0.1	58%	80.51%	56.16%	54%	78.47%	54.11%
0.2	61%	79.02%	57.56%	58%	74.61%	55.18%
0.3	52%	77.20%	53.38%	60%	75.71%	55.38%
0.4	53%	80.89%	52.26%	57%	77.78%	54.26%
0.5	63%	83.53%	57.74%	54%	75.65%	55.66%
0.6	58%	80.06%	55.78%	57%	76.49%	56.01%
0.7	57%	76.48%	56.79%	59%	76.39%	56.13%
0.8	53%	81.39%	55.05%	57%	76.60%	54.82%
0.9	52%	80.86%	50.20%	54%	77.67%	54.78%
1	50%	82.85%	53.10%	55%	76.42%	55.47%
1.1	54%	82.89%	51.66%	55%	76.42%	54.95%
1.2	63%	76.88%	57.13%	59%	79.05%	57.40%
1.3	60%	79.32%	56.92%	52%	77.60%	53.64%
1.4	58%	78.12%	54.99%	56%	74.82%	54.46%
1.5	53%	85.71%	53.02%	52%	76.52%	55.36%
1.6	55%	78.31%	53.16%	59%	76.96%	54.88%
1.7	51%	77.74%	51.91%	58%	77.16%	55.16%
1.8	56%	78.70%	53.60%	59%	76.18%	55.03%
1.9	58%	81.14%	57.85%	54%	75.38%	53.97%
2	65%	79.57%	59.19%	54%	75.58%	52.43%
Average	56.5%	80.0%	54.9%	56.0%	76.6%	54.9%

Figure 5 Llama Test Result

Based on the results in Figure 5, it can be concluded that temperature does not have a significant influence on the classification outcomes. When comparing fixed examples to random examples, the primary observation is a slight increase in voting consistency when using fixed examples. This suggests that while the temperature setting does not heavily impact classification accuracy, the choice of fixed examples over random examples may contribute to more stable and consistent classifications across the model's runs.

- **Test Statistics from GPT-4.o mini**

As mentioned in earlier chapters, GPT's moderation policy is known to prevent the uploading of harmful content that triggers the moderation flag. In smaller tests, such as API calls via Jupyter or UI interactions, warnings for inappropriate content were received, but GPT continued performing the given tasks. However, during the test with 100 data points, some data points were consistently skipped for evaluation in every run. This required bypassing the restriction to continue the test.

Assumption 1: The initial assumption was that the first sentence in the batch flagged the content filter, causing the entire batch to be skipped. However, observations revealed that only a part of the batch in the middle was skipped, which invalidated this assumption.

Assumption 2: Another hypothesis considered was that the content filter might not be triggered by a single data point but by an accumulated threshold for harmful content, where a high percentage of harmful content within the batch could activate the filter. With this hypothesis, the batch size was reduced from 100 to 50, resolving the issue. The results are shown in Figure 6.

Temperature	Fixed			Random		
	Accuracy	Highest Vote average	correct votes average	Accuracy	Highest Vote average	correct votes average
0	57.78%	95.44%	57.44%	55.56%	83.78%	59.28%
0.1	58.89%	95.22%	56.89%	58.89%	82.81%	58.59%
0.2	56.67%	94.33%	56.78%	58.89%	83.99%	60.54%
0.3	56.67%	93.44%	57.28%	58.89%	82.72%	57.63%
0.4	56.67%	91.72%	57.17%	61.11%	79.33%	59.43%
0.5	61.11%	91.00%	58.50%	60.00%	81.21%	60.55%
0.6	56.67%	88.94%	56.83%	67.78%	79.21%	62.27%
0.7	55.56%	87.94%	56.72%	61.11%	80.22%	59.89%
0.8	56.67%	86.10%	57.77%	57.78%	80.54%	59.51%
0.9	55.56%	85.14%	57.42%	62.22%	81.34%	61.27%
1	60.00%	83.41%	57.57%	56.67%	78.49%	58.60%
Average	57.48%	90%	57%	59.90%	81%	60%

Figure 6 GPT-4.o Mini Test Result

The test results showed similar behaviour to the Llama model, with fixed examples providing slightly higher classification consistency. While temperature did not significantly affect the consistency of In-Context Learning (ICL) with random examples, the highest voting average for the fixed examples steadily decreased as the temperature increased.

Awareness of Potential Future Problem: While downsizing the batch size seemed to be a solution, the difference in harmful content between batches of 100 and 50 data points was not thoroughly investigated. Therefore, it remains unclear whether the updated number of 50 is a reliable strategy for the entire test set. To ensure every data point is evaluated in each run strictly, further experiments are required.

7.1.6 Test on the Variance of Model Outputs

In the previous test, GPT-4o-mini achieved an average accuracy of 57.5% for ICL with fixed examples and 59.9% for ICL with random examples. In the test the random example configuration with a temperature of 0.6 reached an Accuracy of almost 68%, a significant outlier. To determine whether this was due to an optimal combination of parameters or a random occurrence, a follow-up test was conducted using GPT4.o Mini with the following setup:

- 100 data points were chosen at random from the dataset.
- The temperature for fixed examples was set to 0.
- The temperature for random examples was set to 0.6.
- Each data point was classified 20 times by GPT.
- Four additional runs were conducted, with an hour-long wait between each run.

GPT was used for this test since Llama didn't have any outliers of this magnitude and no temperature related influence on classification could be confirmed with Lama in the test setup. The temperature of 0 was chosen based on studies such as "The Effect of Sampling Temperature on Problem Solving in Large Language Models" from Renze et al. [31] suggesting that low temperatures are preferred for tasks requiring precision. In the earlier test, no significant difference in accuracy was observed across temperature settings for GPT with fixed example ICL. The temperature of 0.6 was used because it produced the outlier result in the previous test.

Run	Fixed, temperature = 0			Random, temperature = 0.6		
	Accuracy	Highest Vote average	correct votes average	Accuracy	Highest Vote average	correct votes average
1	57.78%	95.44%	57.44%	67.78%	79.21%	62.27%
2	55.56%	95.50%	55.50%	56.67%	86.10%	57.77%
3	58.89%	96.28%	57.39%	58.89%	80.44%	58.93%
4	57.78%	95.78%	57.56%	61.11%	81.21%	59.68%
5	54.44%	94.94%	55.11%	64.44%	80.34%	61.79%
Variance	3.34	0.24	1.42	19.50	7.24	3.63

Figure 7 Variance Test Results

From the results in

Run	Fixed, temperature = 0			Random, temperature = 0.6		
	Accuracy	Highest Vote average	correct votes average	Accuracy	Highest Vote average	correct votes average
1	57.78%	95.44%	57.44%	67.78%	79.21%	62.27%
2	55.56%	95.50%	55.50%	56.67%	86.10%	57.77%
3	58.89%	96.28%	57.39%	58.89%	80.44%	58.93%
4	57.78%	95.78%	57.56%	61.11%	81.21%	59.68%
5	54.44%	94.94%	55.11%	64.44%	80.34%	61.79%
Variance	3.34	0.24	1.42	19.50	7.24	3.63

Figure 7 the sample variance was calculated using the following formula:

$$s^2 = \frac{\sum(x_i - \bar{x})^2}{n - 1}$$

The results of this follow-up test, indicate that the high performance of random examples ICL with a temperature of 0.6 was likely an outlier. This conclusion is supported by the high variance observed across the different runs when using random examples.

7.1.7 Reliability Calculation

Based on the tests described in Section 7.1.5, the probability of a classification belonging to the majority is around 95% for GPT and around 80% for Llama. Given that these probabilities are relatively low for contexts where reliable results are required, the dataset is labelled over multiple iterations. The results of these iterations are then aggregated via a majority vote, which serves as the model's final decision.

To determine the number of iterations required for the majority vote to remain reliable, a binomial distribution was used. In this distribution, *n* is the number of iterations and *Pr* is the probability of a classification being in the majority.

$$P(\mathrm{k}) = \sum_{k=0}^{n} \binom{n}{k} Pr^k (1 - Pr)^{n-k}$$

A threshold of 0.5% is reasonable as it ensures a very low probability of misclassification. To achieve this, the reduction in misclassification per odd iteration is calculated and shown in Figure 8:

Iterations	GPT, 95%	Llama, 80%
1	5%	20%
3	0.725%	10.4%
5	0.116%	5.792%
7	0.019%	3.334%
9	-	1.958%
11	-	1.165%
13	-	0.700%
15	-	0.424%

Figure 8 Probability of Misclassification

Based on these results, it can be concluded that 7 iterations for GPT and 15 iterations for Llama provide the necessary confidence that the classifications are reliable.

7.2 Fine-Tuned Model

To leverage a model with a deeper understanding of the dataset, a fine-tuned model is employed in the majority voting process. While many pre-trained versions of BERT are openly accessible and have demonstrated high performances, using these models could lead to artificially high performance due to overlapping train-test datapoints, since the model might have been trained on the HateXplain dataset. Moreover, an excessively well-performing model may experience diminished effectiveness when its decisions are aggregated with those of less proficient models. Therefore, a self-trained BERT model is used.

As detailed in Section 4.2, BERT is fine-tuned using a training data subset split from the original dataset. This training data, balanced to ensure an equal number of entries for each class, is utilized exclusively for fine-tuning and is not included in the majority voting process, as training models on imbalanced datasets has been shown to yield poorer results in studies, such as the one by Jung et al. [32].

7.2.1 Fine-Tuning Process

The base BERT model used is the BERT-base-uncased model available from Hugging Face [33]. To fine-tune the base model for the classification task with three outputs, the Hugging Face Trainer API was employed to train the model over three epochs. The training data was divided into five folds for cross-validation, with an 80% training set and a 20% validation set.

7.2.2 Fine-Tuning Test

Given BERT's deterministic nature, a preliminary test was conducted where 100 data points were classified 50 times. Each data point consistently received the same label across all iterations, indicating that labelling the dataset only once provides a set of classifications free from randomness for BERT. This approach yielded an accuracy of 62%.

BERT has a maximum input size of 512 tokens, necessitating the processing of each data point individually. In the test, BERT demonstrated the stability and reliability of its deterministic decisions, meaning each point needs to be processed only once.

7.3 The Ensemble

This chapter describes the how the ensemble is constructed and how the bias was calculated.

7.3.1 Output Encoding

GPT and Llama return their predicted answers as text. For the purpose of aggregating the models output classifications, the textual results were transformed into numerical symbols to represent the label outputs. The numbers themselves do not carry any intrinsic numerical meaning; each one corresponds to a specific binary decision for a data entry:

0 represents the case "normal = True"

1 represents the case "offensive = True"

2 represents the case "hate = True"

For example: Input: “text” → Output: 2, meaning the input is classified as hate speech by the model.

- **Considerations for Confidence / Probability**

The Confidence Score, or Classification Threshold, indicates how sure the NLP service / machine learning model is that the respective intent was correctly assigned. A Confidence Score is a probability between 0 and 1 that represents the likelihood that the output of a machine learning model is correct and will satisfy a user’s request. If the confidence level falls below a predefined limit, a fallback intent is issued.

LLMs like GPT or Llama do not inherently provide confidence scores for their classifications. These models are designed to generate text based on patterns and context, rather than explicitly calculating probabilities or confidence scores for specific outputs. Using an external model as a confidence estimator could be an option to deliver result in confidence. Without expanding the project scope by introducing another tool, one could use the fine-tuned BERT model as the confidence estimator. However, this would shift the decision-making process from being parallel (i.e., independent) to dependent. Therefore, the majority voting in this context becomes unfair. For these reasons, the confidence score is not adopted in this project.

- **Considerations for Using Vectors or One-Hot Encoding**

One-Hot Encoding is a method for converting categorical variables into a binary format by creating new binary columns (0s and 1s) for each category in the original variable. In machine learning models, One-Hot Encoding helps eliminate ordinality and enables models to capture complex relationships within the data, converting categorical variables into a suitable format.

To maintain consistency with the BERT model, which outputs a single label per datapoint, the model was restricted to assigning only one label per datapoint. This approach ensures that the classification strategy remains aligned with BERT's capabilities and expectations, simplifying the decision-making process and avoiding potential conflicts with multi-label classification.

```
Original data :
   ID  Label  Prediction  Result
0  10      0           1   False
1  20      1           1    True
2  15      1           0   False
3  25      2           2    True
4  30      1           2   False

Encoded data :
   ID  Label_0  Label_1  ...  Prediction_2  Result_False  Result_True
0  10      1.0      0.0  ...           0.0           1.0          0.0
1  20      0.0      1.0  ...           0.0           0.0          1.0
2  15      0.0      1.0  ...           0.0           1.0          0.0
3  25      0.0      0.0  ...           1.0           0.0          1.0
4  30      0.0      1.0  ...           1.0           1.0          0.0
```

Figure 9 Encoding Example

Figure 9 illustrates how the encoding approach would be applied in this case. Using vectors or one-hot encoding does not provide additional useful information in this specific scenario.

- **Consideration for percentage-based aggregation**

A percentage-based approach has been considered for aggregating the results of the iterations, either by using the proportion of votes directly or applying a confidence threshold. However, the majority vote was selected, as it provides a binary output label consistent with the annotations in the dataset.

This approach involves trade-offs, such as the loss of certain details in contested cases (for example, if Llama voting 7/8 leads to 7 votes being disregarded). Nevertheless, the simplicity and ease of comparison to the ground truth outweigh these drawbacks, particularly given the limited time available for this study.

7.3.2 Final Majority Vote:

In simulating the original dataset label decision strategy, each data point is assigned one final decision by every LLM. Each decision carries equal weight in the final vote, representing a hard voting mechanism. Given that there are three voters and three legitimate decision options, a tie situation is possible. The impact of such tie situations on the overall voting results from the dataset cannot be predetermined and will be considered during the result analysis.

Figure 10 illustrates the confusion matrix design for individual LLMs and the ensemble:

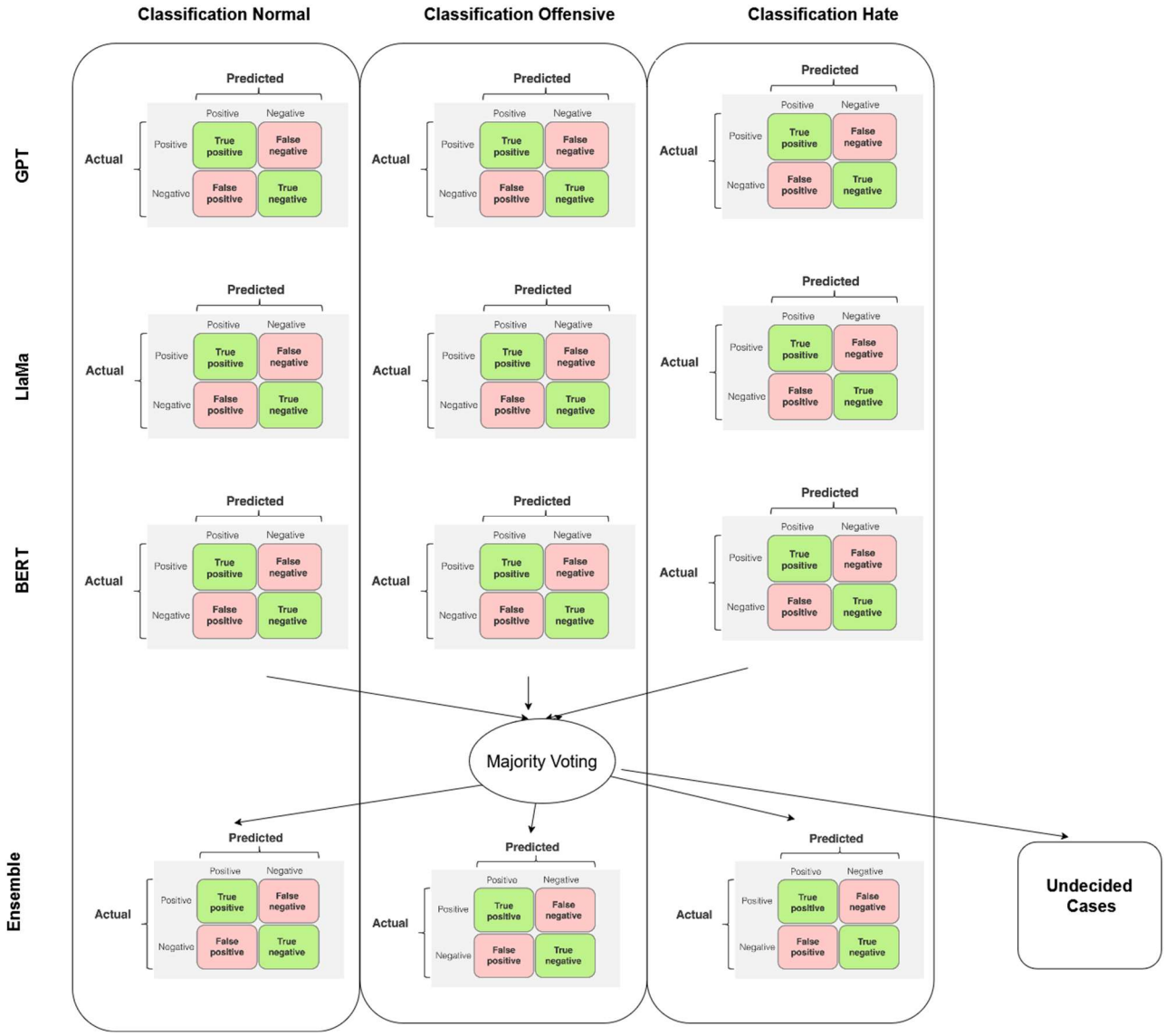


Figure 10 Confusion Matrix of Individual LLM and Ensemble

- **Consideration for Outliers**

Outliers occur in cases where one or two LLMs exhibit more bias, especially if such examples are not chosen for learning in the prompt. The democracy of voting may lead to poor decisions in these instances. These situations exist in the real world. This experiment simulates a real-world voting approach without assuming or deciding which LLM has more knowledge or power before actual performance measurement.

- **Consideration for Boosting Strategy**

Boosting is an ensemble learning technique that aims to improve the performance of machine learning models by combining the outputs of several weak learners to create a strong learner for more reliable output. Boosting tends to have less effect when used on strong learners. The models used in this study, including LLMs using ICL and fine-tuned BERT, are not overly strong performers, as a more noticeable difference compared to ensemble results is expected to be observed. However, since these models are not weak learners either, boosting is not adopted in this study.

7.4 Measuring Bias Reduction

To avoid imposing any arbitrary order on the categories (such as ranking "Hate" > "Offensive" > "Normal"), the Hamming distance is employed. This metric is considered well-suited for nominal data as it calculates the "distance" between categories by determining whether the predictions are correct or incorrect, without assuming any inherent order between the categories. This means the distance between any two labels is treated as equal. Misclassification is calculated using integer counts, where each misclassified instance contributes one unit of distance, resulting in the formula depicted in Figure 11. This essentially results in the opposite of the overlap measure discussed in this paper from Boriah et al. [34].

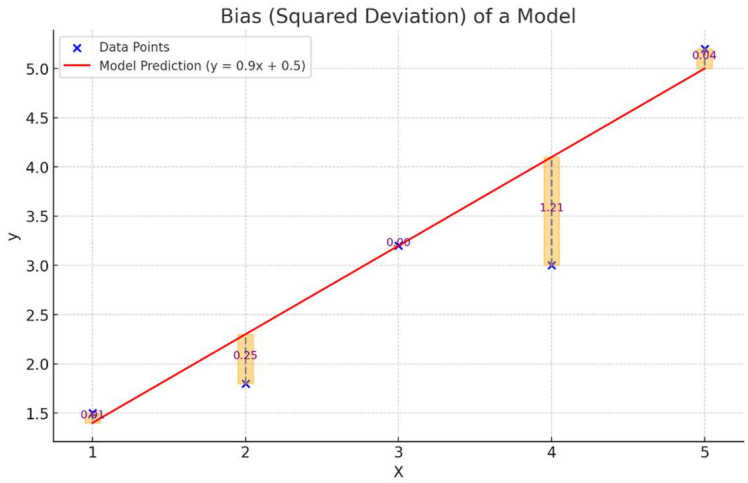
$$\text{Bias} = \sum(\text{incorrect predictions})$$

Figure 11 Bias Unit

Therefore, this approach provides a simple and effective way to measure the extent of bias in the model's outputs, as it quantifies the number of misclassifications and does not introduce any unwanted ordering of categories.

- **Consideration for Standard Deviation**

In machine learning, bias is commonly quantified by the squared deviation of the model's predictions from the actual data points an example of which is depicted in Figure 12. This deviation reflects how "wrong" the model's predictions are relative to the true labels in the test data. However, this standard approach is not suitable for nominal data, as is the case in this study.



$$\text{Bias} = (-0.10)^2 + (0.50)^2 + (0.00)^2 + (1.10)^2 + (-0.20)^2 = 1.51$$

Figure 12 Bias Calculation using Standard Deviation

Nominal data does not lend itself to traditional numerical distance calculations, such as Euclidean or Manhattan distances, which are designed for continuous or ordinal data. These metrics cannot be directly applied to categories that are simply labels, with no inherent order or numerical relationship between them. Additionally, since the categories in this study are one-dimensional and represented solely by textual labels (e.g., "Hate," "Offensive," "Normal"), it is not feasible to use common similarity measures like Jaccard or cosine similarity. Therefore, the use of standard deviation in this context would not provide meaningful results, as it relies on assumptions of numerical distances that do not apply to nominal data.

- **Limitations of Alternative Methods for Measuring Prediction Bias**

While alternative methods such as the Chi-squared test for goodness of fit and Cramér's V are commonly employed for analysing categorical data, they are not ideal for measuring prediction bias. These tests primarily evaluate whether the observed frequency distribution of categories matches the expected distribution, rather than directly assessing whether the model's predictions are correct or incorrect. As a result, they do not offer direct insights into prediction errors, which are essential for understanding bias in a classification model.

For example, consider a situation where the model makes incorrect predictions for all samples, as illustrated in the table below:

	Correct Label	Model Prediction
100 samples	Hate	Normal
100 samples	Offensive	Hate
100 samples	Normal	Offensive

$$E(\text{Hate}) = E(\text{Offensive}) = E(\text{Normal}) = \frac{100 + 100 + 100}{3} = 100$$

$$\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i} = \frac{(-100)^2}{100} + \frac{(0)^2}{100} + \frac{(-100)^2}{100} + \frac{(-100)^2}{100} + \frac{(-100)^2}{100} + \frac{(0)^2}{100} = 400$$

In this case, although the model has made incorrect predictions for every data point, the Chi-squared test still returns a result that confirms that the model's predictions align with the expected distribution, misleadingly implying that the model performs acceptably. This is because the Chi-squared test does not directly evaluate the correctness of individual predictions but rather compares overall frequencies.

Thus, while the Chi-squared test may indicate that the model's predictions do not significantly deviate from the expected distribution, it fails to capture the true performance of the model, particularly in cases where all predictions are incorrect. In such cases, the test may erroneously suggest that the model is performing adequately. Therefore, the Chi-squared test is suboptimal for directly measuring bias reduction in classification tasks, as it does not assess the accuracy of the model's individual predictions.

8. Experiment Implementation

After evaluating various factors, the implementation of the experiment followed the decisions outlined in the previous chapters. Specifically, the experiment is conducted with 7 runs using a temperature setting of 0 for GPT, 15 runs with a temperature setting of 0.8 for Llama and a single evaluation for BERT. For each large language model (LLM), the model's decision is determined by the most frequently predicted label across all runs.

8.1 Confusion Matrix for Individual LLMs

The confusion matrices below summarize the performance of each model.

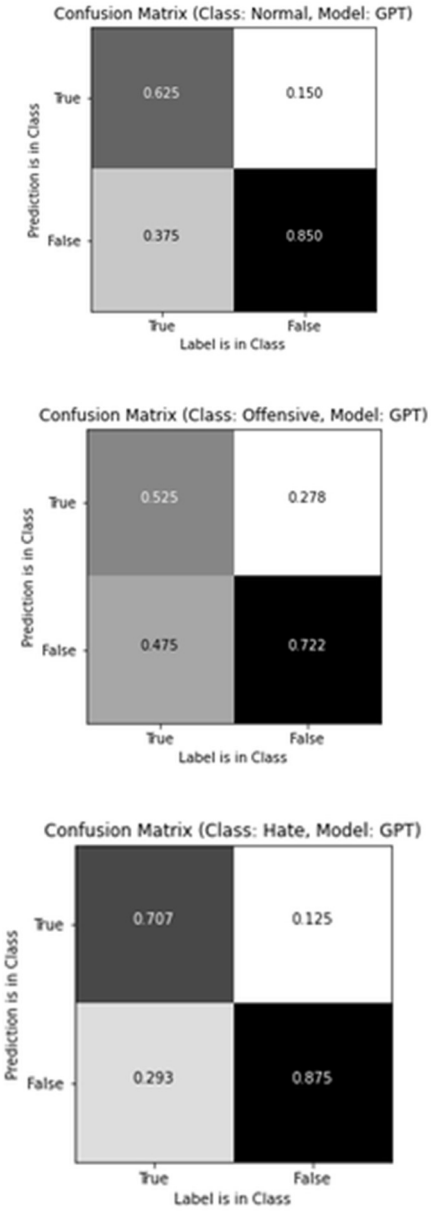


Figure 13 GPT Confusion Matrices

GPT: The confusion matrices for GPT in Figure 13 displays the accuracies for each label as follows: "Normal" with 62.5%, "Offensive" with 52.5% and "Hate" with 70.7%.

Figure 14 Llama Confusion Matrices

Llama: the confusion matrices for Llama in Figure 14 shows the following accuracies: "Normal" with 59.7%, "Offensive" with 45.4% and "Hate" with 69.5%.

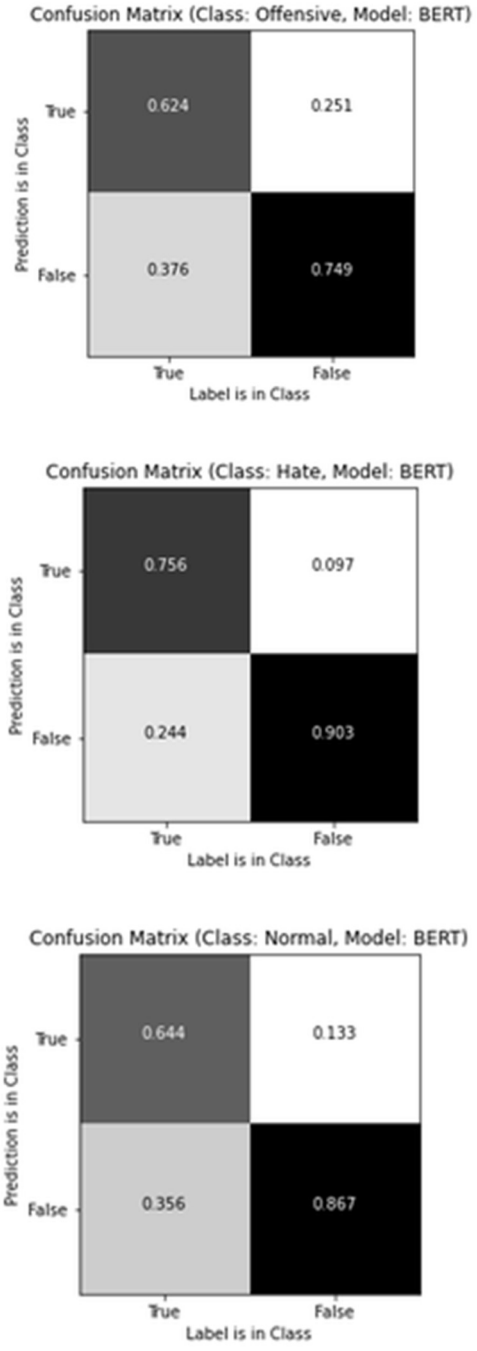


Figure 15 BERT Confusion Matrices

BERT: The confusion matrices for BERT in Figure 15 displays the accuracies as follows: "Normal" with 64.4%, "Offensive" with 62.4% and "Hate" with 75.6%.

The performance of BERT appears superior in predicting every category, particularly in the category “Offensive”, where BERT shows notably higher accuracy than the other models.

8.2 Tie in Majority Voting

Notably, 714 tie cases are identified during majority voting, representing approximately 5.4% of the total test set size. These tie cases are distributed across the following labels: 369 instances labelled as "Normal," 199 instances labelled as "Offensive," and 146 instances labelled as "Hate." The distribution of these undecided cases is depicted in Figure 16. It appears that BERT has the highest chance of being correct in such cases.

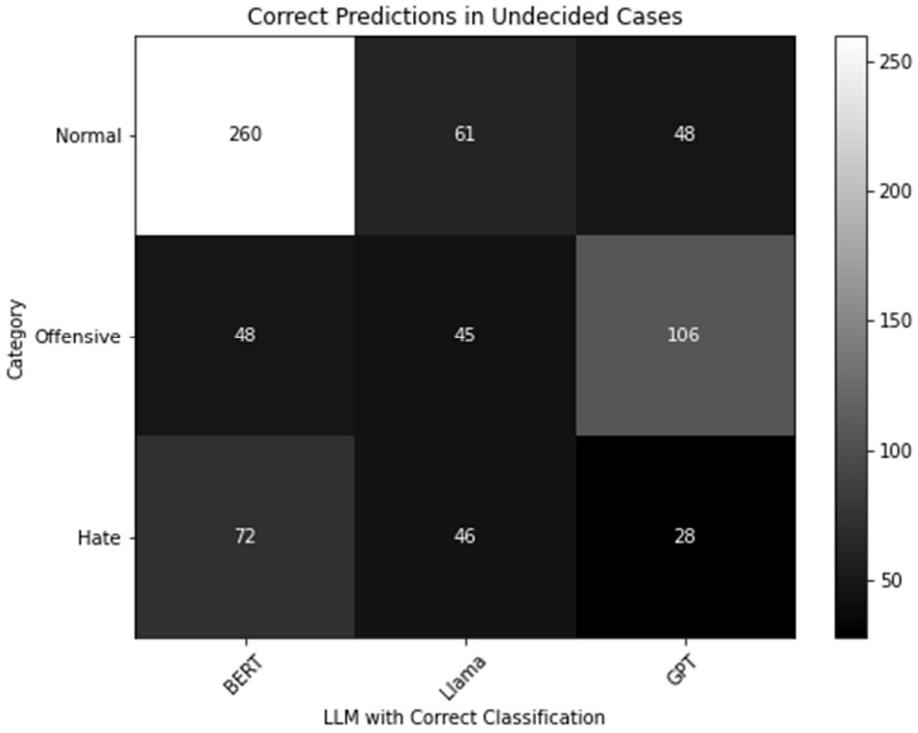


Figure 16 Correct Prediction in Undecided Cases per Model

For these ambiguous cases, two separate approaches are being considered: either excluding them from the evaluation (which is referred to as Exclusion Method) or replacing them with the decision made by the best-performing model (which is referred to as Replacement Method). The impact of these approaches on bias calculation and ensemble performance will be examined in detail in the following chapters.

8.3 Bias Reduction Comparison

Building on the statistical results obtained earlier, the performance of individual LLMs can be compared with the two ensemble methods discussed in previous sections. The former individual LLM analysis identified BERT as the best-performing model for the Replacement Method.

8.3.1 Total Bias Changes

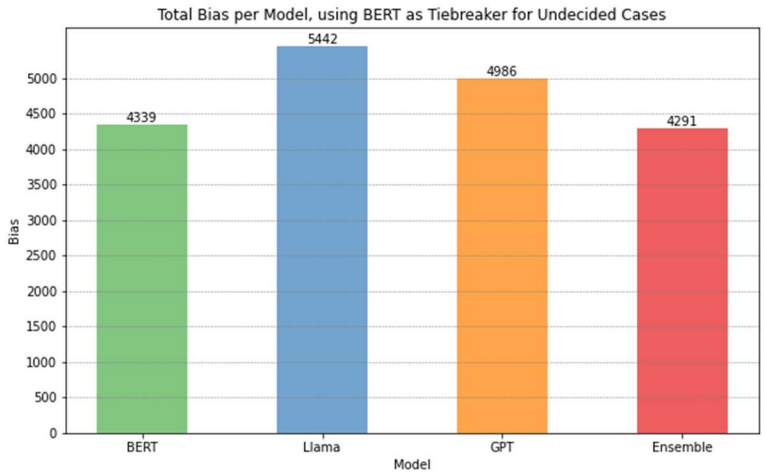


Figure 17 Total Bias, Replacement Method

The results depicted in Figure 17 reveal that the ensemble method reduced bias by approximately 1% compared to the best individual model, which is BERT. Additionally, the ensemble reduced bias by around 14% when compared to GPT and 21% when compared to Llama. These findings suggest that the ensemble method can effectively reduce bias, even if the reduction is modest when compared to the best-performing single model. A similar behaviour can be observed using the exclusion method as depicted in Figure 18.

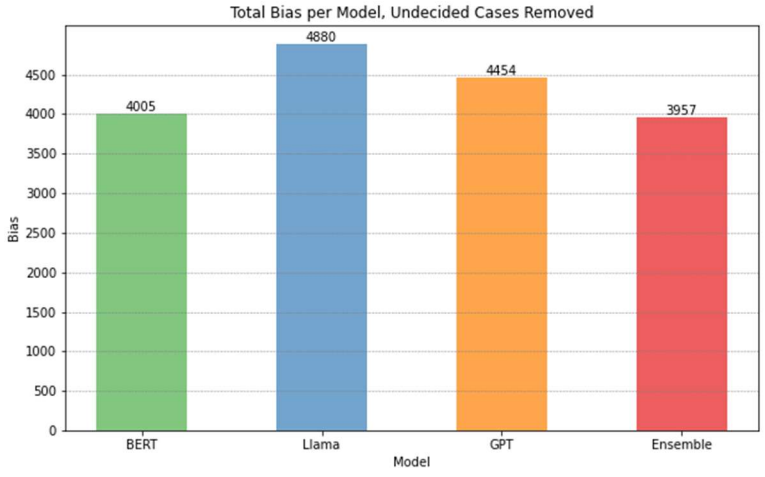


Figure 18 Total Bias, Exclusion Method

8.3.2 Directional Bias Tendency Changes

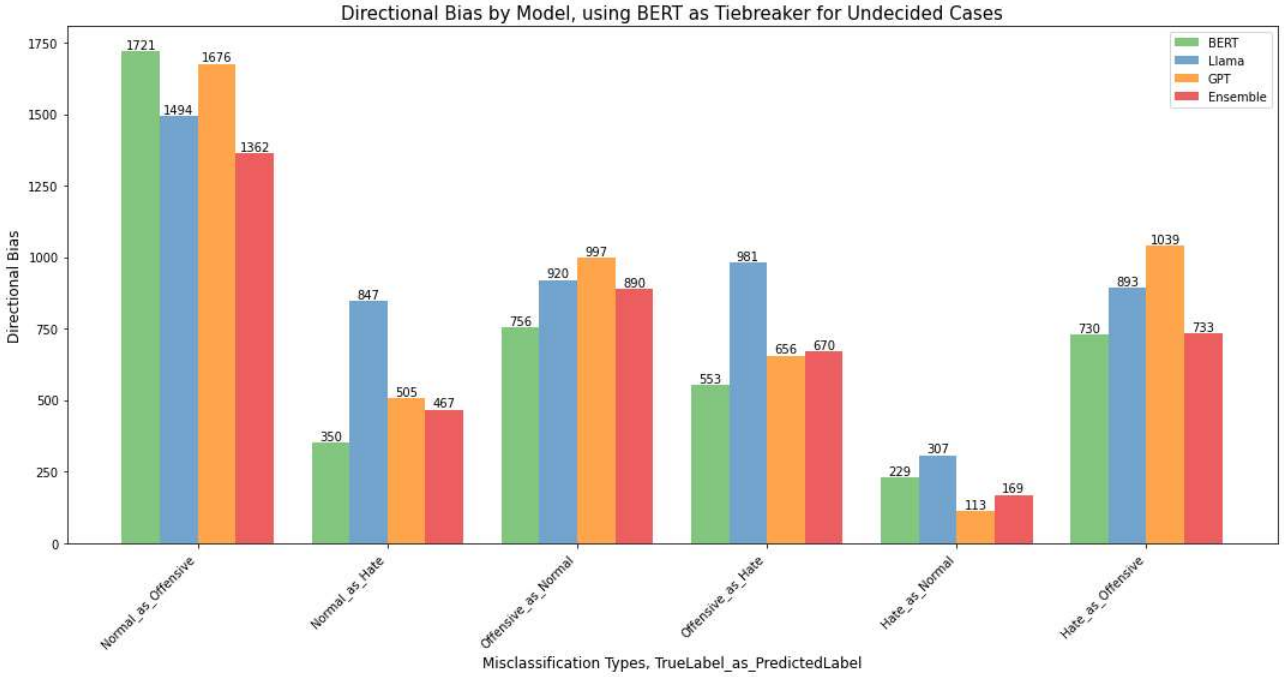


Figure 19 Directional Bias Tendencies, Replacement Method

Figure 19 illustrates the extent and direction of the bias, as described in Chapter 5. Since bias is defined in this paper as the number of misclassified data points, the maximum bias in any direction corresponds to the number of data points assigned to the true label of that direction. As outlined in Chapter 6, the test dataset includes 3’935 posts labelled as Hateful, 3’480 as Offensive and 5’814 as Normal, which represent the maximum bias for the respective categories. Upon reviewing these results, it becomes evident that the ensemble method introduced slightly greater bias than BERT in most directions. However, it outperformed BERT in specific cases, such as misclassifying hate speech as normal and normal speech as offensive, with fewer misclassifications in these directions.

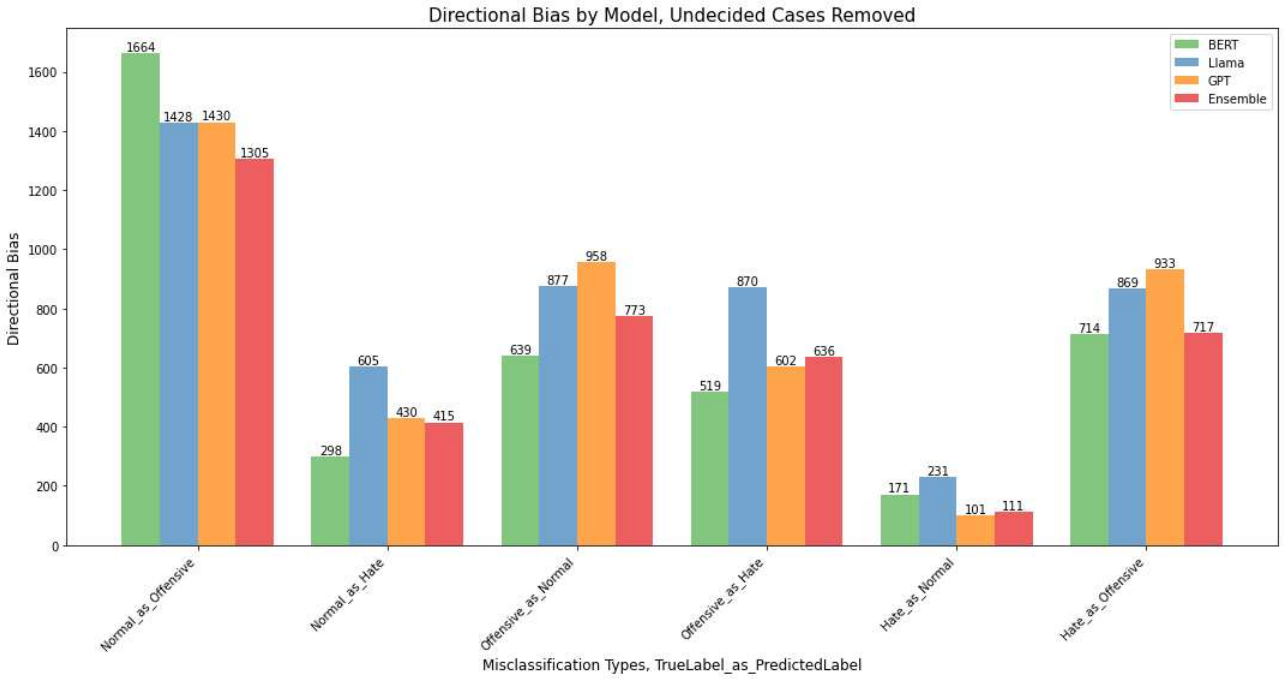


Figure 20 Directional Bias Tendency, Exclusion Method

As depicted in Figure 20 the exclusion of the undecided cases resulted in some of the models weaknesses being mitigated. For example, GPTs tendency to categorize samples from the normal category as offensive is more in line with Llama and the ensemble using this method.

8.3.3 Accuracy Changes

Although accuracy was not the primary focus of this study, the accuracy of the models is presented in Figure 21 and Figure 22 to provide a broader understanding of their performance across the entire dataset, as the bias highlighted in the previous chapters only represents a portion of it. This is important, as bias reduction can sometimes manifest in a slight increase in accuracy rates.

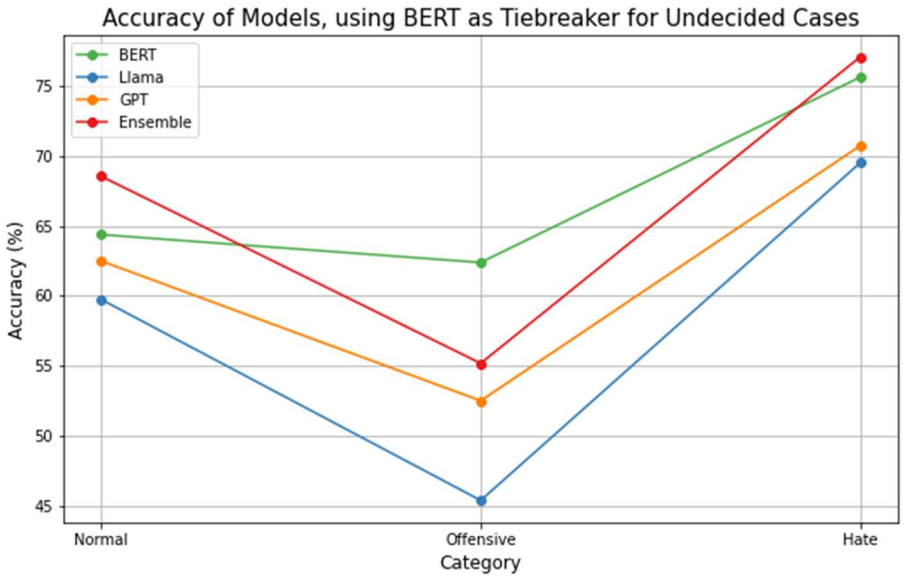


Figure 21 Accuracy, Replacement Method

The analysis of Figure 21 indicates that the ensemble approach performs optimally when the participating models exhibit similar levels of accuracy, particularly in the "Normal" and "Hate" speech categories. However, when individual model performance varies significantly, as seen in the "Offensive" category, the ensemble method is outperformed by the best-performing single model.

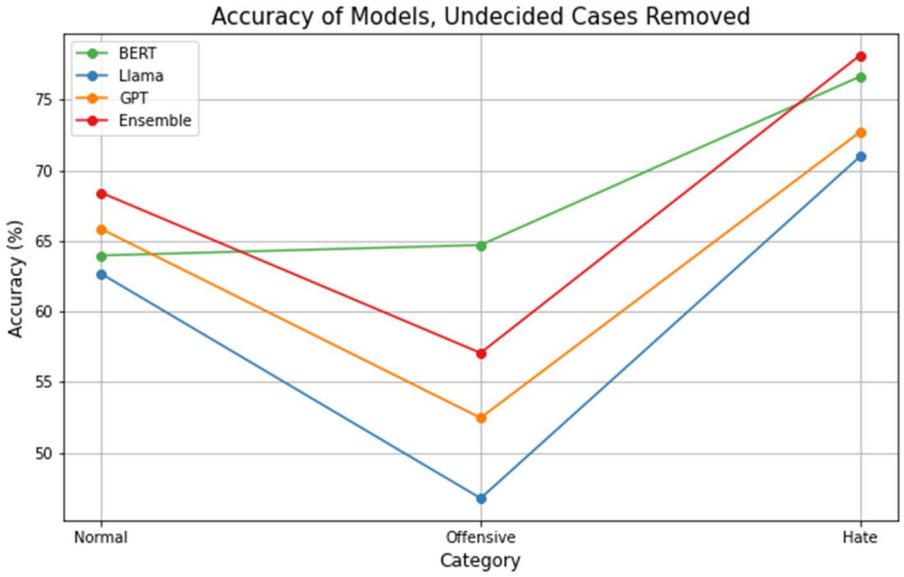


Figure 22 Accuracy, Exclusion Method

Notably, the Exclusion Methods accuracy presented in Figure 22 slightly outperforms the Replacement Method. This is expected, since the undecided cases represent the most ambiguous cases. Excluding them from the evaluation leads to better overall performance.

8.4 Cost Based Analysis for Ensemble Decisions

While both ensemble methods discussed in the previous sections have certain mathematical limitations and drawbacks, their effectiveness largely depends on the specific context and the goals of the application. In some scenarios, where the principle of "a wrong answer is better than no answer" applies, the Exclusion Method (removing undecided cases) may not be feasible. In other contexts, such as healthcare, where a false negative could allow a patient with a contagious disease to freely move in public, this scenario could be considered far more costly than a false positive, where the patient stays in the hospital for further examination. In such cases, the Replacement Method needs to be evaluated carefully to ensure its appropriateness.

To generalize the evaluation of ensemble methods, a mathematical/statistical framework is required to assess the reliability of each approach with respect to a defined goal. This framework should apply across various datasets and similar task settings. A cost-based analysis is suggested to us by Andreas Müller. The following chapter uses the methods described in section 14.B of his book “Lineare Algebra: Eine anwendungsorientierte Einfuhrung” [35] as a foundation. The formulations used can also be found in his book [35, p. 722].

8.4.1 Transitional Matrix (T)

The evaluation process begins by considering the probabilities P(classified as X | actually Y), i.e., the probability that an item is classified as X ∈ { normal, offensive, hate } when it is actually Y. This forms a probability matrix that describes the probabilistic outcomes of the classification method. The matrix is constructed as follows:

$$P = \begin{bmatrix} \text{correct\_normal} & \text{normal\_as\_offensive} & \text{normal\_as\_hate} \\ \text{offensive\_as\_normal} & \text{correct\_offensive} & \text{offensive\_as\_hate} \\ \text{hate\_as\_normal} & \text{hate\_as\_offensive} & \text{correct\_hate} \end{bmatrix}$$

Each element represents the number of instances of misclassification, then the columns are normalized to L1-norm 1, ensuring that the matrix represents a probability matrix.

8.4.2 Frequency Vector (p)

In addition to the transition matrix, the frequency of each category in the dataset is captured in a vector p:

$$p = \begin{bmatrix} \text{true\_normal} \\ \text{true\_offensive} \\ \text{true\_hate} \end{bmatrix}$$

This vector is then normalized, converting it into a probability vector representing the distribution of the categories in the dataset.

8.4.3 Cost Matrix (C)

To assess the cost of misclassifications, a cost matrix C is introduced. This matrix has the same shape as the transition matrix and assigns a cost to each type of misclassification. Diagonal elements represent the cost of correct classifications (which may be negative, to reward correct classifications which is typically desirable). Off-diagonal elements reflect the cost of misclassifications. A cost matrix could be structured as follows:

$$C = \begin{bmatrix} \text{cost\_normal\_normal} & \text{cost\_normal\_offensive} & \text{cost\_normal\_hate} \\ \text{cost\_offensive\_normal} & \text{cost\_offensive\_offensive} & \text{cost\_offensive\_hate} \\ \text{cost\_hate\_normal} & \text{cost\_hate\_offensive} & \text{cost\_hate\_hate} \end{bmatrix}$$

8.4.4 Cost Calculation

Once the transition matrix *T* and the cost matrix *C* are defined, the expected cost of classification can be computed. The elementwise product (Hadamard product) of the cost matrix and the transition matrix yields a matrix containing the risk of each type of misclassification. Further multiplying the risk matrix by the frequency vector *p*, returns a vector containing the cost incurred for each classification label, which can be calculated as:

$$(C \odot T)p$$

The total cost can be obtained by summing the elements of the resulting vector or multiplying by a column vector of ones *U*, which serves as a unit element in the Hadamard product:

$$U \cdot (C \odot T)p$$

This value represents the expected cost of classification and can be used to compare different classification methods according to the costs they incur. Additionally, this approach allows for the projection of cost changes if the distribution (represented by the frequency vector p) changes, as the cost is a linear function with coefficient vector $U \cdot (C \odot T)$.

8.4.5 Case Study and Cost Calculation Demonstration

To demonstrate the utility of the cost-based evaluation method, consider a use case in forum moderation, where misclassifying hate speech as normal is considered far more severe than misclassifying normal speech as hateful or offensive. In this case, undecided cases are excluded from the dataset, as they represent edge cases that should be reviewed by a human moderator. After exclusion, the dataset shrinks to 12,515 datapoints, with 43.5% classified as normal, 26.2% as offensive and 30.3% as hate speech.

The distribution of categories is captured in the frequency vector *p*:

$$p = \begin{bmatrix} 0.435 \\ 0.262 \\ 0.303 \end{bmatrix}$$

For this example, the misclassification costs are set as follows:

- Normal classified as Hate speech or Offensive: Cost = 1
- Hate speech classified as Offensive (or vice versa): Cost = 1
- Hate speech or Offensive classified as Normal: Cost = 1.5

This cost structure reflects the severity of certain misclassifications, treating a misclassification of inappropriate content as normal as more severe. The resulting cost matrix for this example is depicted in Figure 23:

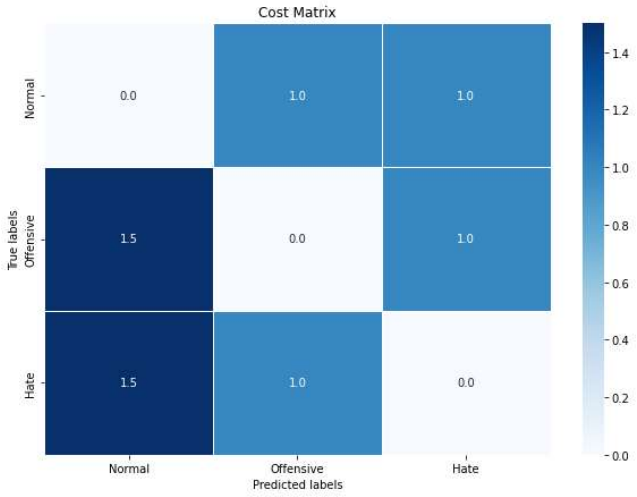


Figure 23 Cost Matrix

With these matrices in place, the expected cost for each model can be computed using the formula:

$$ExpectedCost = U \cdot (C \odot T)p$$

Using these values, these are the expected costs for each model:

- BERT: 0.398
- Llama: 0.471
- GPT: 0.430
- Ensemble: 0.385

These results indicate that, under the selected cost matrix, the Ensemble method incurs slightly less cost than BERT, suggesting it provides a better balance between classification accuracy and cost reduction, particularly in the context of severe misclassifications.

9. Conclusion

The results show that an ensemble can reduces bias and improve accuracy in labelling hate speech and normal language compared to individual models. However, in the category “Offensive”, the ensemble did not outperform the best single model, indicating that the hypothesis did not hold in this case. For ambiguous cases, where the majority voting leads to a draw (around 5.4% of 13’229 entries), using the exclusion method improves accuracy and reduces bias more effectively than using the replacement method. However, the usage of the exclusion method will require additional validation methods as it leaves some entries undecided. This study highlights the potential of an ensemble approaches to enhance fairness and accuracy in hate speech detection systems.

9.1 Personal Reflections

Team Member Kailing Peng

Embarking on this hate speech detection project has provided me with deep insights into NLP AI models, particularly the three models that are used in this study: GPT-4o mini, Llama 3.1 8B using ICL and fine-tuned BERT. Working with these models has enhanced my understanding of their strengths, limitations and the intricacies involved in their implementation.

For this study, we dived deep into research to gain a comprehensive understanding of bias in LLMs and the biases discussed in the hate speech detection field. One important takeaway is the diversity of bias tendencies, which vary from model to model, dataset to dataset and task to task. Additionally, I took the chance to explore how static and dynamic factors could affect the performance of LLMs, especially with ICL guiding the LLMs' decision-making process.

This project also underscored the importance of addressing biases in LLM models. By examining the biases inherent in our models, I gained a better understanding of how these biases can influence outcomes and the necessity of mitigating them to ensure fairness and accuracy.

Moreover, the process of fine-tuning, evaluating and comparing model performance against ensemble methods was particularly insightful. It demonstrated how even models that are not considered weak learners could benefit from ensemble techniques.

Overall, this project has been an exciting experience, enhancing my technical skills and deepening my understanding of ethical considerations in AI. It has motivated me to continue exploring the field of NLP and AI, with a focus on developing fair and effective solutions for complex social issues like hate speech detection. This project has also made me more careful and sceptical when implementing solutions using LLMs.

Team Member Lukas Derungs

Since I didn’t really read any academic writing before this or learned how to properly write or reason in such an environment, I found the creation of a study such as this to be pretty difficult. The notations are precise and everything must be validated by a trusted source. The best part about working on this study was when I got to work with the models and conduct the tests. From setting up the experiment to analysing the results, each steps had obstacles which means considerations that had to be made and compared by their individual drawbacks and merits. Every step from research to execution to evaluation made me learn much more about LLMs and academic writing than any lecture could have thought me.

10. Open Questions and Recommendations

This study experimented with an ensemble approach of LLMs for hate speech detection. Due to time and resource constraints, several potential improvements were not explored but warrant further investigation:

1. **Multi-Label Classification**: Allowing models to assign multiple labels to a single datapoint could provide a richer understanding, particularly for cases that are both offensive and hateful. For example, an entry like "you are such…" could be labelled as [0,1,1] (normal, offensive, hate). A percentage-based aggregation of model decisions could refine the final prediction (e.g., {0 = 33.3%, 1 = 66.6%, 2 = 0%}). This approach could offer a more nuanced interpretation of the data, but due to time constraints, it was not explored in depth in this study.
2. **Chain of Thought Prompts**: Research suggests that prompting models to generate reasoning before deciding could improve outcomes in hate speech detection. Studies like “Designing of Prompts for Hate Speech Recognition with In-Context Learning” highlight the potential benefits of this approach, which could offer transparency in model decision-making, especially when majority voting fails. Asking multiple LLMs to explain their reasoning might also help reach a consensus when conflicts arise.
3. **Sample Weights**: A limitation of this study is the fixed selection of training examples. Future work should explore dynamic sample selection based on test instance characteristics. Furthermore, the unequal distribution of categories in the test set could impact the model's performance. Using class distribution-based sampling could enhance the model's ability to learn from a more representative set of examples.
4. **Model Weights**: If the accuracy of participating models is known in advance, a weighted voting system could be implemented, allowing for the best model to have more votes compared to the weaker models, increasing its influence on the output.

As the field evolves, continued experimentation and refinement of these approaches will be essential in improving the accuracy, reliability and ethical considerations of AI models in sensitive domains like hate speech detection.

11. APPENDIX A: Literature List

1. Bundesamt für Statistik. Widespread use of the internet is making the Swiss population more vulnerable to fake news and hate speech. Available: Link
2. Soral W, Bilewicz M, Winiewski M. Exposure to hate speech increases prejudice through desensitization. Aggress Behav. 2018 Mar;44(2):136-146. Available: Link. Epub 2017 Nov 2. PMID: 29094365.
3. Arcila Calderón, C., Sánchez Holgado, P., Gómez, J. et al. From online hate speech to offline hate crime: the role of inflammatory language in forecasting violence against migrant and LGBT communities. *Humanit Soc Sci Commun* **11**, 1369 (2024). Available: Link
4. Karsten Müller, Carlo Schwarz. Fanning the Flames of Hate: Social Media and Hate Crime, *Journal of the European Economic Association*, Volume 19, Issue 4, August 2021, Pages 2131–2167. Available: Link
5. Ona de Gibert, Naiara Perez, Aitor García-Pablos and Montse Cuadros. Hate Speech Dataset from a White Supremacy Forum. In Proceedings of the 2nd Workshop on Abusive Language Online (ALW2), pages 11–20, Brussels, Belgium. Association for Computational Linguistics. L. Available: Link
6. MacAvaney S, Yao H-R, Yang E, Russell K, Goharian N, Frieder O (2019) Hate speech detection: Challenges and solutions. PLoS ONE 14(8): e0221152. Available: Link
7. Thomas Davidson, Dana Warmsley, Michael Macy, Ingmar Weber. Automated Hate Speech Detection and the Problem of Offensive Language. Available: Link
8. Çağrı Çöltekin. A Corpus of Turkish Offensive Language on Social Media. In Proceedings of the Twelfth Language Resources and Evaluation Conference, pages 6174–6184, Marseille, France. European Language Resources Association, 2020. Available: Link
9. Cambridge dictionary, Available: Link
10. Collins dictionary, Available: Link
11. Law insider dictionary, Available: Link
12. B. Mathew, R. Dutt, P. Goyal, A. Mukherjee, "Spread of Hate Speech in Online Social Media," Indian Institute of Technology, Kharagpur, 2018. Available: Link
13. Matos, E., Tomaz, R., Maia, A., Sanches, D., Bentes , A., Foletto, L., & Santos, L. (2024). Online hate speech and instant messaging apps: An emerging research agenda. *First Monday*, *29*(9). Available: Link
14. D. Rieger, A. S. Kümpel, M. Wich, T. Kiening, G. Groh, "Assessing the Extent and Types of Hate Speech in Fringe Communities: A Case Study of Alt-Right Communities on 8chan, 4chan and Reddit," 2021. Available: Link
15. F. J. Aranda Serna, "Memes as Symbols of Hate Speech: The Influence of Graphic Humour on Freedom of Expression and Politics," 2024. Available: Link
16. European Commission and IT Companies, "Code of Conduct on Illegal Online Hate Speech," May 31, 2016. Available: link
17. Network Enforcement Act. Available: [Link]
18. GOV.UK, "Online Safety Act: Explainer," May 8, 2024. Available: Link
19. Keshav Kumaresan, K. Vidanage. 2019 National Information Technology Conference (NITC). HateSense: Tackling Ambiguity in Hate Speech Detection, Available: Link
20. Simona Frenda, The role of sarcasm in hate speech. A multilingual perspective. Available: Link
21. Lingrui Mei, Shenghua Liu, Yiwei Wang, Baolong Bi, Xueqi Cheng, SLANG: New Concept Comprehension of Large Language Models , Available: Link
22. Kapelańska-Pręgowska J, Pucelj M. "Freedom of Expression and Hate Speech: Human Rights Standards and Their Application in Poland and Slovenia". Laws. 2023; 12(4):64. Available: Link
23. 1. Siegel AA. Online Hate Speech. In: Persily N, Tucker JA, eds. *Social Media and Democracy*. SSRC Anxieties of Democracy. Cambridge University Press; 2020:56-88. Available: Link

24. A. Das, Z. Zhang, N. Hasan, S. Sarkar, F. Jamshidi, T. Bhattacharya, M. Rahgouy, N. Raychawdhary, D. Feng, V. Jain, A. Chadha, M. Sandage, L. Pope, G. Dozier, C. Seals, "Investigating Annotator Bias in Large Language Models for Hate Speech Detection," arXiv, Jun. 2024. Available: Link
25. R. Pan, J.A. García-Díaz, R. Valencia-García, "Comparing Fine-Tuning, Zero and Few-Shot Strategies with Large Language Models in Hate Speech Detection in English," CMES - Computer Modeling in Engineering and Sciences, vol. 140, no. 3, pp. 2849-2868, 2024. Available: Link.
26. B. Mathew, P. Saha, S.M. Yimam, C. Biemann, P. Goyal, A. Mukherjee, "HateXplain: A Benchmark Dataset for Explainable Hate Speech Detection," arXiv, Dec. 2020. Available: Link
27. T. Garg, S. Masud, T. Suresh, T. Chakraborty, "Handling Bias in Toxic Speech Detection: A Survey," arXiv, Feb. 2022. Available: Link
28. B. Mathew, P. Saha, S.M. Yimam, C. Biemann, P. Goyal, A. Mukherjee, "HateXplain: A Benchmark Dataset for Explainable Hate Speech Detection," AAAI, 2021. Available: Link
29. M. Zhang, J. He, T. Ji, C.-T. Lu, "Don't Go To Extremes: Revealing the Excessive Sensitivity and Calibration Limitations of LLMs in Implicit Hate Speech Detection," arXiv, Feb. 2024. Available: Link
30. K. Guo, A. Hu, J. Mu, Z. Shi, Z. Zhao, N. Vishwamitra, H. Hu, "An Investigation of Large Language Models for Real-World Hate Speech Detection," arXiv, Jan. 2024. Available: Link
31. Matthew Renze, Erhan Guven, The Effect of Sampling Temperature on Problem Solving in Large Language Models, Available: Link
32. Vincent Jung, Lonneke van der Plas, Understanding the effects of language-specific class imbalance in multilingual fine-tuning, Available: Link
33. Huggingface, Bert-base-uncased, Available: Link
34. Shyam Boriah, Varun Chandola and Vipin Kumar, Proceedings of the 2008 SIAM International Conference on Data Mining, Similarity Measures for Categorical Data: A Comparative Evaluation, Available: Link
35. Müller, A. (2023). Positive Matrizen. In: Lineare Algebra: Eine anwendungsorientierte Einführung. Springer Vieweg, Berlin, Heidelberg, Available: Link
36. Karl Weinmeister, Beyond temperature: Tuning LLM output with top-k and top-p: Link

12. APPENDIX B: Graphics

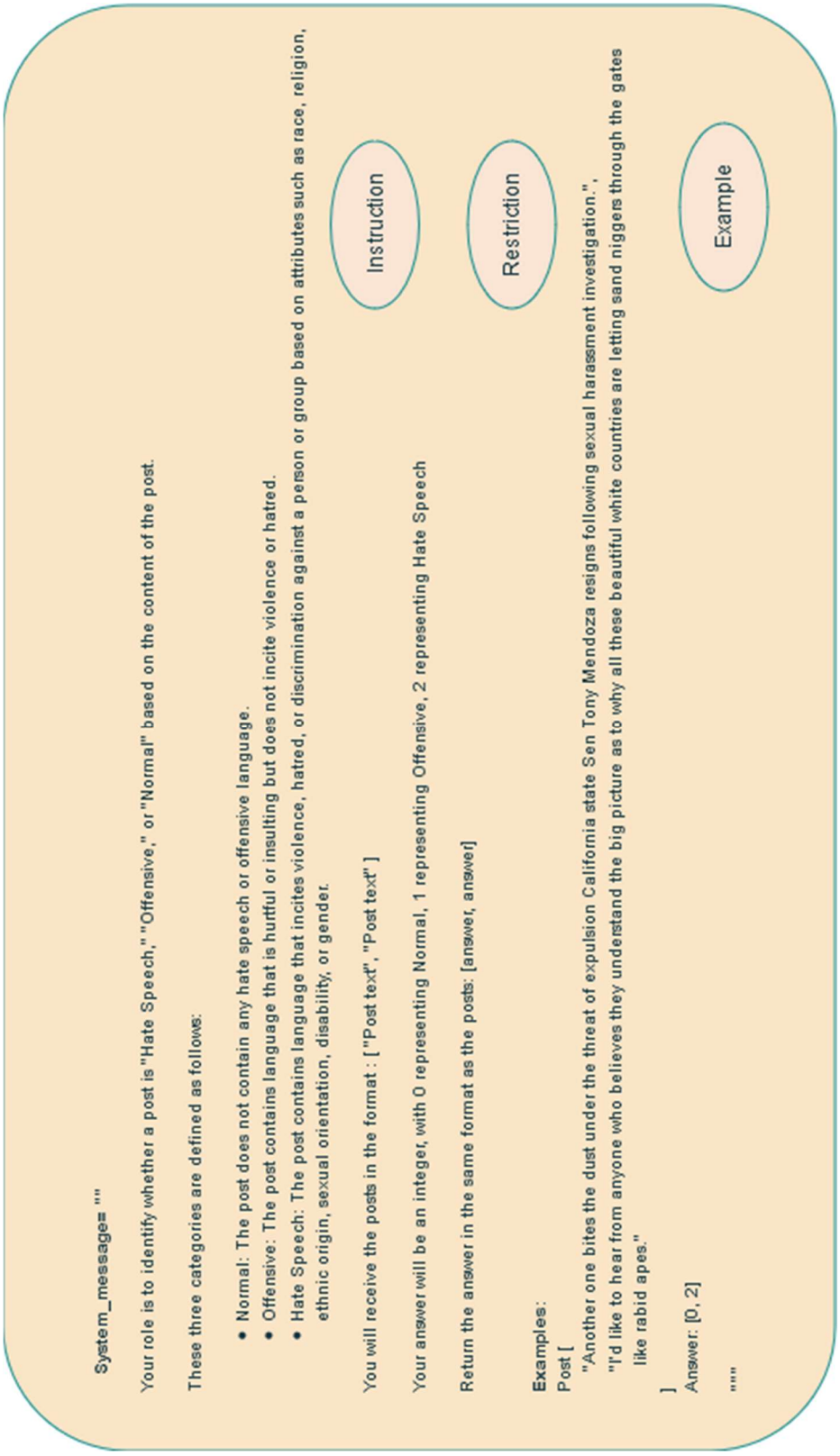


Figure 24 Prompt Design Example