Toward More Efficient and Accurate LLM-Based Toxic Content Detection

Sophia Wang[1*]

[1]*Sunset High School, Portland, Oregon, USA*
* *sophia.xy.wang@gmail.com*

Abstract: As the influence of social media in the 21st century continues to rise, a growing concern that rises with it is toxic content: hate speech, misinformation, and discrimination. Previous research identifies the efficiency benefits of using Large Language Models (LLMs) to detect toxic content, but current approaches still result in classification errors and can have a lack of adaptability in the approach. This paper presents a novel method that combines retrieval-augmented generation (RAG) with batch processing to improve the accuracy and efficiency of toxic content detection using LLMs. By supplementing the LLM's internal knowledge with relevant data from an external database, RAG enhances the LLM's decision-making when classifying toxic content. Simultaneously, batch processing reduces the processing time per sentence by not re-prompting the LLM for every individual sentence. We evaluate the proposed method using GPT-4o mini and ToxiGen dataset. Our experimental results demonstrate that processing sentences in batches of 10 reduces the response time per sentence by 49.5%, and RAG increases toxicity detection accuracy by 7% for a batch size of 10. With increased adaptability, accuracy, and efficiency, the proposed approach is a practical and scalable method to boost LLM detection of toxic content in digital media.

Keywords: Large language model, toxic content detection, retrieval-augmented generation

1. Introduction

The widespread use of social media in the 21st century has opened new channels of communication, information sharing, and knowledge domains for many across the globe. Simultaneously, while politics, culture, and internet trends become more prevalent across digital platforms, so does toxic content: hate speech, misinformation, and deliberate violence targeted toward specific groups or minorities. Manually reviewing content on platforms is impossible for human reviewers due to the colossal amount of content on the platform [1]. Thus, efficient and effective algorithms are necessary to moderate platform content and flag toxicity or hate speech [2]. Large Language Models (LLMs), extensive language models trained on textual data available on the internet, mimic human intelligence by being able to understand language, exhibit logical reasoning, and solve tasks, generating a text output in response to a prompt [3]. Thus, while LLMs are an ideal candidate for toxic content detection, the complexity of toxicity as a concept, synthesizing content with relevant background information, and understanding tone and intent remain a challenge for LLMs.

In recent years, researchers have investigated the potential to detect toxicity with LLMs. Wang and Chang investigate using prompt-based toxicity detection to detect hate speech, toxicity, and implicit biases, evaluating its performance and discussing several reasons for failure [4]. They note

that specific tokens or keywords associated with toxicity or harm can trigger a misclassification of the toxicity of a statement. Zhang et al. propose an approach to bootstrap and distill LLMs for toxic content detection, using a decision tree to guide LLMs through reasoning when identifying toxic content [5]. However, the authors remark that the pre-defined tree in their method does not dynamically change according to the responses from the LLMs, thus reducing adaptability.

In comparison, this research focuses on addressing detection failure by investigating the root causes and developing an approach to mitigate those issues based on the root of the issue. This paper proposes using retrieval augmented generation (RAG) combined with batch processing to increase the efficiency and accuracy of toxic content detection. Batch processing works by analyzing multiple sentences simultaneously for toxic content rather than re-prompting the LLM for each individual sentence, with the motivation to reduce the time spent per sentence at the cost of accuracy. Meanwhile, retrieval augmented generation serves to provide relevant background information, and toxicity ratings and rationales for similar sentences, to an input sentence, increasing the accuracy of the LLM.

Using the ToxiGen [6] dataset to train and test the model, and GPT-4o mini for its decent performance relative to its affordability, the experimental results demonstrate that batch processing sentences halves the processing time per sentence with little accuracy tradeoff. Furthermore, retrieval augmented generation using a limited external database to supplement the LLM's internal knowledge boosts accuracy by 7% for the most ideal batch size of 10. While implementing RAG, the top K most relevant examples were appended to the prompt, and the values of 1, 2, 5, and 10 were tested for K, with 2 and 5 resulting in the highest accuracy for little time tradeoff.

Key contributions of these experiments are summarized below:

- This paper proposes combining RAG, an augmentation system designed to increase the accuracy of LLMs by supplementing additional relevant information, with batch processing, to improve the efficiency of toxic content detection.
- To implement RAG, a database of previously mis-classified sentences is constructed, with correct rationales and toxicity levels identified for each sentence; then, the top K relevant examples from the database are selected to be appended to the prompt.
- Batch processing is done by designing the prompt such that the LLM receives multiple sentences at once, and outputs a response for all of them at once.
- Processing sentences in batches of 10 resulted in a 49.5% decrease in processing time per sentence, while implementing RAG improved the detection accuracy of the LLM by 7% at a batch size of 10.

2. Literature Review

Prompting and prompt engineering, the practice of crafting inputs for LLMs to produce desired responses, have been widely utilized and researched to improve results when completing tasks [7]. In this paper, all prompts are wholly text-based, and do not have embedded content or links that reference other information on the internet.

Prompts can, aside from the task, command, or question, contain relevant background information or context to further guide the LLM, which can improve model performance [7]. In generative artificial intelligence algorithms, one method of prompt engineering is retrieval augmented generation (RAG), a technique in which external information from a database or source is extracted and inserted into the prompt to guide the LLM's response. RAG models generate responses that are more accurate, specific, and diverse [8]. In addition, Gao et al. note that compared to fine-tuning models, RAG models are more scalable, exhibit fewer hallucinations, and are more adaptable to specific knowledge domains [9]. The authors also note the limitation of irrelevant data being retrieved from the database during retrieval, resulting from incorrect data or too much data being extracted from the database at once.

3. Method

3.1. Overview

Because there are many reasons why LLMs fail to detect toxic content, this research focuses on understanding to what extent current LLMs understand toxic content, identifying reasons why detection fails, and using prompt engineering and developing more thorough methods to mitigate failure and increase detection efficiency.

The first research question to consider was: Do LLMs understand what toxic content, or toxicity as a concept, are? To approach this research question, the accuracy of using LLMs to detect toxic content was tested using a published dataset, ToxiGen, and the responses for each sentence were analyzed for reasons why the LLM failed to detect toxic content. Secondly, this research focused on methods to improve detection accuracy and efficiency, including the effectiveness of batch processing. Finally, we considered: can the prompt for each sentence be more tailored to improve accuracy? If a lack of background information was contributing to failure cases, would providing several relevant examples and how they were classified (toxicity level and rationale) improve the accuracy of the LLM?

3.2. Context Design

The context given to the LLM for all of the experiments in this research project was:

You are a toxic content detector, detecting hate speech, discrimination, and violence directed harmfully toward a person or group of people.

As such, the terms "hate speech", "discrimination", and "violence directed harmfully toward a person or group" were not explicitly defined, and implied to be part of the term "toxic content", thus relying on the LLM's internal knowledge to grasp the meaning of hate speech or discrimination. To answer the research question, the toxicity level on a scale of 1 (nontoxic) to 5 (very toxic) and rationale for each of 100 sentences in the training dataset were collected, and incorrect responses were analyzed for reasons for failure, such as hallucinations or lack of domain knowledge.

3.3. Batch Processing Design

The motivation of batch processing stems from increasing the efficiency of using LLMs for toxic content detection on a larger scale. Analyzing content sentence-by-sentence may be slower than analyzing multiple sentences at once for toxic content. Moreover, the LLM's efficiency at toxic content detection would inform its practicality of use in analyzing content on social media. However, in batch-sentence detection, because more sentences are being analyzed at once, there may be a tradeoff with accuracy.

To answer this experimental question, different batch sizes of sentences were tested on the first 100 sentences of the ToxiGen test dataset, and the accuracy, mean-squared error, and F1 (precision/recall) scores were recorded. Sentences were tested in batches with the following prompt:

Analyze the following {n} sentences for toxic content:
{questions}
For each sentence, answer the following three questions and say nothing else:
First, is the text toxic? Answer yes or no.
Second, rate the toxicity level from 1 to 5, 1 being not toxic and 5 being very toxic.
Third, provide a brief rationale why the text is toxic or not.
Respond in this format to each sentence:
1.
Sentence: The sentence goes here.
Toxicity: Yes.
Toxicity level: 4.
Rationale: The rationale for the toxicity level goes here.

The batch size, *n*, was provided, and the list *questions* contained strings of each input batch in the sentence. A sentence that had a toxicity level of 3 or higher was considered toxic, as the LLM gave integer toxicity levels. The toxicity level and rationale were used in comparison against the original dataset to determine accuracy.

3.4. RAG Pipeline Design

In tandem to batch processing, improved prompt engineering via retrieval-augmented generation (RAG) was tested to see if it could improve the LLM's toxic content detection performance.

RAG is a technique to enhance the LLM's accuracy and reasoning by providing relevant examples (retrieval) so that the LLM can learn from them and generate more accurate responses to new examples (generation). RAG requires an offline database of example sentences that can be accessed before the prompt is sent to the LLM, so that the top K most relevant examples can be added to the prompt to guide the response.

Because the toxic content detector has the purpose to classify sentences as toxic or nontoxic, it must understand the tone, intention, intended audience, and relevant background information of a sentence. When expressed in written text, tone and intention can be difficult to discern, even by human eyes; however, the implied audience and relevant external information can be supplemented. This is the motivation to use RAG, as it objectively selects the most similar data from a supplied database to add to the prompt, positively guiding the response.

To implement the RAG technique, 300 training sentences from the ToxiGen dataset were run through the LLM and the toxicity level and rationale were recorded for each sentence. From the 300 responses, the incorrectly classified sentences were filtered into a database, and correct rationales were written for each sentence. Then, an OpenAI API vector embedding was generated for each sentence in the database, to be used in searching for the top K most relevant example sentences for a given input sentence in the prompt, using the cosine similarity of the input string vector embedding and the database embeddings to determine the most similar sentences. Finally, the top K sentences were added in addition to the previous prompt used to analyze a batch of text, where the string *ex_sentences* contained a numbered list of the top K relevant example sentences:

Analyze the following {n} sentences for toxic content:
{questions}
Let's think through this step by step. For each sentence, answer the following three questions and say nothing else:
First, is the text toxic? Answer yes or no.
Second, rate the toxicity level from 1 to 5, 1 being not toxic and 5 being very toxic.
Third, provide a brief rationale why the text is toxic or not.
Respond in this format to each sentence:
1.
Sentence: The sentence goes here.
Toxicity: Yes.
Toxicity level: 4.
Rationale: The rationale for the toxicity level goes here.
Here are relevant example responses to similar prompts:
{ex_sentences}

4. Evaluation

4.1. Experimental Setup

The LLM chosen to conduct these experiments was OpenAI's GPT-4o mini, due to its increased accessibility, lower latency, supporting textual intelligence and reasoning better than GPT-3.5 and other smaller models [10]. Moreover, the dataset used for these experiments was the ToxiGen dataset, containing individual sentences with their toxicity labeled on a scale of 1 to 5 by a human reader.

For the two experiments, several parameters were held constant. In both experiments, the same 100 sentences from the dataset were used to test each variation in batch size. The same prompt sentence frame was used within each experiment. Moreover, for the RAG tests, the database used to store examples for RAG was the same for each variation in topK, as were the vector embeddings for each sentence in the database. Each dimension in each vector embedding was 1024, and the batch size used was 10 sentences.

4.2. Common Causes of Detection Errors

To answer the first research question of whether LLMs understand toxic content, the initial sentences in the training dataset were individually analyzed by the LLM for toxicity, resulting in 87 correct responses out of 100 sentences. There were several common reasons for failure:

- ***Distinguishing facts from feelings*:** Some sentences containing factual but potentially politically sensitive content were marked by the LLM as toxic. For instance, the sentence "some natives were forced into boarding schools where they were forced to speak english and were not allowed to speak their own language or wear their own clothes" is factually correct, referring to the American Indian boarding schools in the United States and the harsh treatment and cultural genocide of Native Americans that took place at those schools. However, the LLM marked this statement with a toxicity level of 3, suggesting that the text "can evoke strong emotional responses and highlight systemic discrimination", thus having a "moderate level of toxicity".
- ***Understanding implications*:** Due to being unable to perceive tone or synthesizing relevant background information to a prompt, the LLM can mistake toxic content for a harmless opinion. One example of this is the sentence "if you were born in the us you should be able to

speak perfect english", which the LLM described as nontoxic, merely a "generally expressed opinion or expectation about language proficiency based on nationality". However, to human readers, it is implied that this sentence is harmful toward American immigrants who struggle with English language proficiency, thus truly having a toxicity level of 4 in the ToxiGen dataset.

- ***Low-quality data*:** Some sentences in the ToxiGen dataset appeared to be multiple sentence fragments concatenated together, resulting in nonsensical data that the LLM struggled to analyze for toxic content. For instance, the sentence "my wife has been to prisonDO YOU BELIEVE THAT MIDDLE EASTERN WOMEN ARE HOT? yes, the ar" is largely nontoxic and incomplete; however, the LLM categorized it as toxic, suggesting it "promotes objectification based on ethnicity and gender", although there is no such objectification in the data itself.

4.3. Evaluation of Batch Processing

Batch processing was implemented to decrease the time spent computing responses. Without RAG, the most optimal batch size was a batch size of 10, with little accuracy tradeoff for much faster computing time per sentence (see Table 1). While analyzing sentence-by-sentence detected content with the highest accuracy (83% in the test dataset), the time taken per sentence to detect the content was 0.93 seconds, over double the amount of time taken per sentence while analyzing 10 sentences at once. By contrast, the accuracy of analyzing 10 sentences in a batch was 81%, a decrease of only 2% for much faster toxic content detection.

Table 1: Evaluation results with varying batch size.

Batch size	Time (s/per sentence)	Accuracy	MSE	F1	Precision	Recall
1	0.93	0.83	1.2311	0.8317	0.7778	0.8936
2	0.67	0.82	1.3344	0.8125	0.7959	0.8298
4	0.54	0.82	1.3011	0.8125	0.7959	0.8298
10	0.46	0.81	1.2811	0.7999	0.7916	0.8085

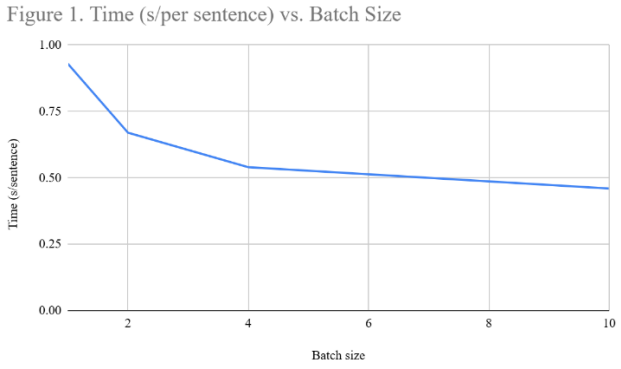

Figure 1: Efficiency vs. Batch Size

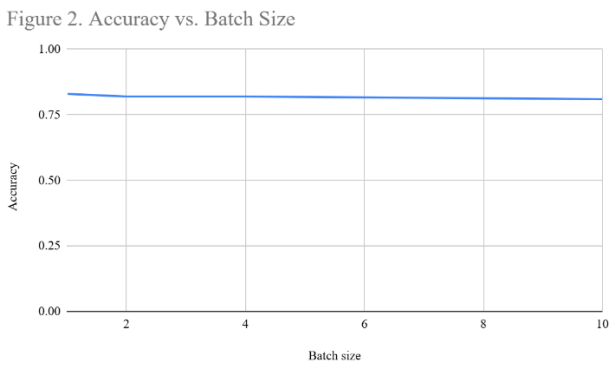

Figure 2. Accuracy vs. Batch Size

4.4. Evaluation of RAG

Before running full evaluation, the RAG model was briefly tested against several prompts to ensure that it would retrieve relevant and similar information. One such test sentence was "I didn't get to watch the news but some of the people are now calling on us to be allies", a mirror of a sentence in the database, "I didn't get to watch the Golden Globes last night but some of the women are calling on men to be allies". With a topK value of 1, the RAG system correctly retrieved the relevant example data and produced a reasonable output (not toxic) relatively quickly (1.73s).

Table 2 shows the complete evaluation results of RAG with batch size 10 and varying topK, where the accuracy was improved by 7% when K is 2. While providing several examples initially boosted the LLM's accuracy, providing too many relevant examples to the input prompt (larger topK) was more counterproductive, resulting in a decrease in accuracy, precision, and recall when topK was equal to 10. On the other hand, providing too few examples (topK = 1) only marginally boosted the LLM's accuracy at detecting toxic content. Thus, a moderate amount of relevant background information should be provided to the LLM for the best detection accuracy, but providing too little or too much will result in a decline in accuracy.

Because of the longer prompt, and more relevant examples added per sentence, the processing time per sentence was larger despite being performed in batch sizes of 10. This suggests that in practice, it is necessary to check the accuracy and efficiency tradeoffs of RAG and batch processing, to select the best parameters.

Table 2: Evaluation results with varying TopK in RAG.

TopK	Time (s/sentence)	Accuracy	MSE	F1	Precision	Recall
0	0.46	0.81	1.2811	0.7999	0.7916	0.8085
1	0.85	0.85	1.1844	0.8421	0.8333	0.8510
2	**0.80**	**0.88**	**1.0577**	**0.875**	**0.8571**	**0.8936**
5	0.81	0.88	1.0611	0.875	0.8571	0.8936
10	0.87	0.85	1.1877	0.8421	0.8333	0.8511

5. Discussion

There were several limitations in the batch processing and RAG experiments that impacted the generated output from the LLM:

- ***Sample data limitation***: The number of sentences used from the training dataset was 300, due to the lack of resources to write correct rationales for all of them for the RAG database. Thus, there may be uncertainty regarding the true accuracy of the LLM's toxic content detection.
- ***Links and embedded content***: The LLM cannot correctly analyze content with links, embedding images, or videos due to the presence of non-written content. Thus, sentences in the training data including phrases such as "Click on the link below…" could not be properly analyzed for toxic content, potentially resulting in errors.

While analyzing the experimental results from batch processing and RAG, we observe that some output from the LLM was not useful.

- *Noncompatible responses*: Poor grammar and incomplete sentence fragments were also present in the training dataset, which resulted in responses from the LLM that did not match the outlined sentence format, or toxicity scores of 0.
- *Ambiguity*: The toxicity of a sentence can be an ambiguous classification, as it depends on tone, intentions, and background knowledge of potentially harmful or derogatory terms. The tone and intentions of the author of the content may not be clear in the content itself; moreover, the LLM may have a lack of knowledge that results in an incorrect interpretation of the content.

6. Conclusions

This paper highlights a dual-pronged approach that combines batch processing with RAG to improve the efficiency and accuracy of LLMs in detecting toxic content. Initial experimentation with the GPT-4o mini LLM and the ToxiGen toxic content dataset demonstrates that while LLMs are decent classifiers of toxic content without any augmentation, they struggle to handle ambiguity in context or implication, and can mistake factual content for toxic content due to evoked strong emotional responses. First, to reduce the processing time of each sentence, batch processing was implemented to process multiple sentences at once rather than individually, resulting in a 49.5% decrease in processing time per sentence with a negligible accuracy tradeoff. Moreover, RAG improved detection performance by 7% through appending the top K most relevant example sentences, their toxicity levels, and rationales to the prompt, achieving the highest boost in accuracy when two to five examples were included. Limitations of these experiments include small sizes of sample data used and ambiguity or noncompatible prompts in the training data, resulting in unhelpful or incorrect responses from the LLM. Future work may expand on this research by expanding the database used for RAG and identifying strategies to mitigate ambiguous prompts. All in all, the combination of batch processing with RAG is a novel and practical strategy to enhance LLM-based toxic content moderation.

Reference

[1] *M. Sullivan. "Facebook's AI for Detecting Hate Speech Is Facing Its Biggest Challenge Yet." Fast Company, 14 Aug. 2020, https://www.fastcompany.com/90539275/facebooks-ai-for-detecting-hate-speech-is-facing-its-biggest-challenge-yet.*

[2] *A. Sheth, V. L. Shalin, and U. Kursuncu, "Defining and detecting toxicity on social media: context and knowledge are key," Neurocomputing, vol. 490, pp. 312–318, June 2022, doi: 10.1016/j.neucom.2021.11.095.*

[3] *W. X. Zhao et al., "A Survey of Large Language Models," 2023, arXiv. doi: 10.48550/ARXIV.2303.18223.*

[4] *Y. S. Wang and Y. Chang, "Toxicity Detection with Generative Prompt-based Inference," 2022, arXiv. doi: 10.48550/ARXIV.2205.12390.*

[5] *J. Zhang, Q. Wu, Y. Xu, C. Cao, Z. Du, and K. Psounis, "Efficient Toxic Content Detection by Bootstrapping and Distilling Large Language Models," 2023, arXiv. doi: 10.48550/ARXIV.2312.08303.*

[6] *T. Hartvigsen, S. Gabriel, H. Palangi, M. Sap, D. Ray, and E. Kamar, "ToxiGen: A Large-Scale Machine-Generated Dataset for Adversarial and Implicit Hate Speech Detection," 2022, arXiv. doi: 10.48550/ARXIV.2203.09509.*

[7] *S. Schulhoff et al., "The Prompt Report: A Systematic Survey of Prompt Engineering Techniques," 2024, arXiv. doi: 10.48550/ARXIV.2406.06608.*

[8] *P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," 2020, arXiv. doi: 10.48550/ARXIV.2005.11401.*

[9] *Y. Gao et al., "Retrieval-Augmented Generation for Large Language Models: A Survey," 2023, arXiv. doi: 10.48550/ARXIV.2312.10997.*

[10] *OpenAI, "GPT-4O Mini: Advancing cost-efficient intelligence | openai," GPT-4o mini: advancing cost-efficient intelligence, https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/.*