Post++

Social Media Moderation and Correction Built Using LLMs

ECE1786H F Progress Report

Dec 08, 2024

Word Count: 1931
Word Count Penalty: 0%

Chao-Lin Chen - 1009582186

David Zhang - 1003260918

Mingchen Du - 1009716611

Yi-Fan Hung - 1010807799

0. Permissions

Name	Post Video	Post Final Report	Post Source Code
Chao-Lin Chen	Yes	Yes	Yes
David Zhang	Yes	Yes	Yes
Mingchen Du	Yes	Yes	Yes
Yi-Fan Hung	Yes	Yes	Yes

1. Introduction

Post++ is an automated User Generated Content (UGC) moderation tool designed to enhance the integrity and safety of social media platforms by leveraging Large Language Models (LLMs). Current approaches involve tool-assisted, human moderation of social media, which is labor-intensive, prone to bias, and struggles to scale with the volume of content generated daily. Conversely, many fully-automated moderation systems struggle to capture nuanced language or sarcasm, which often require sophisticated reasoning capabilities. By utilizing LLMs, Post++ seeks to address these gaps while maintaining a scalable and consistent moderation process. This system evaluates multimodal UGC against platform-specific Community Guidelines (CG), identifying violations and suggesting constructive edits to align the content with these policies.

2. Illustration / Figure

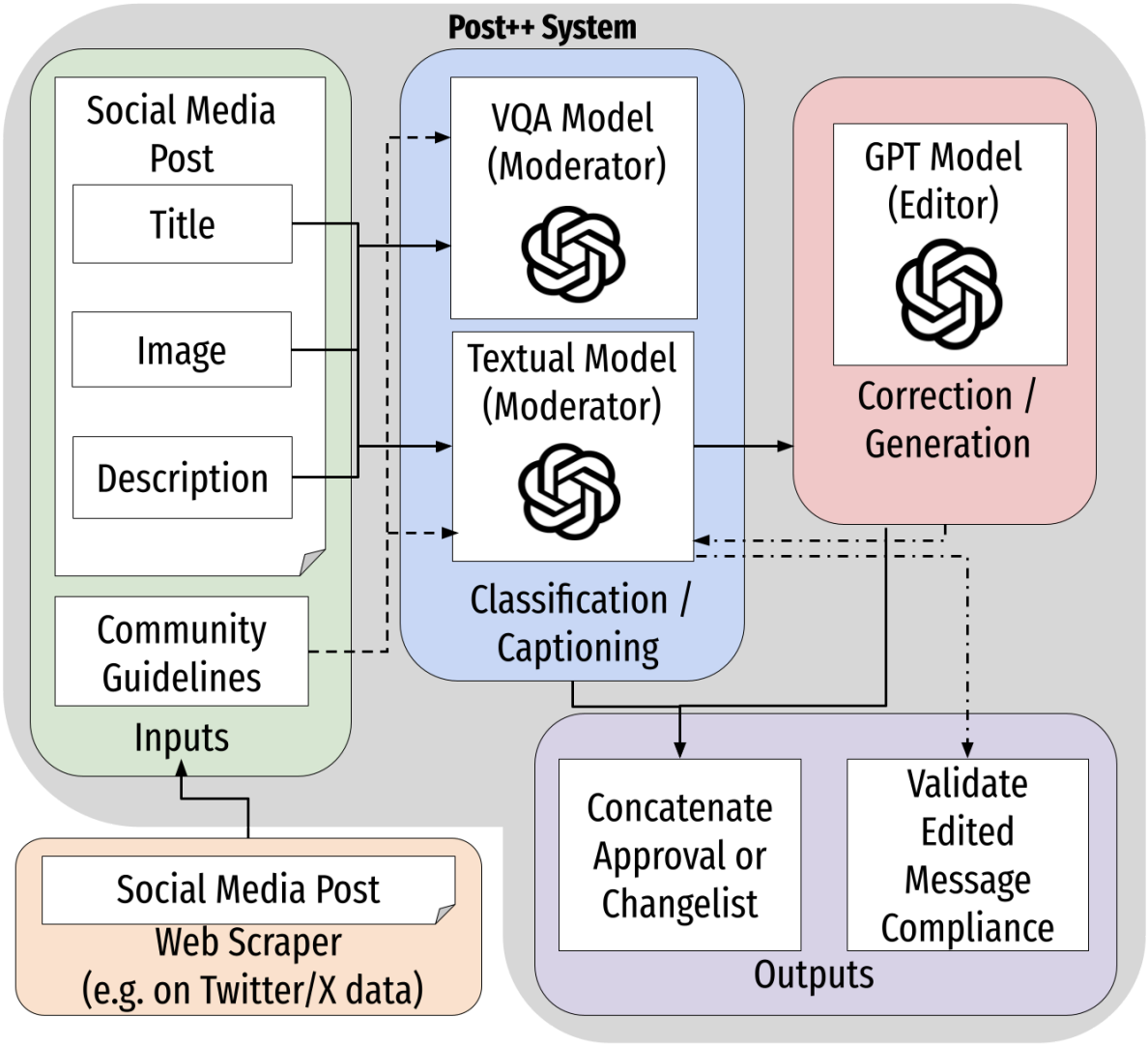


Figure 1: System architecture diagram for Post++

3. Background & Related work

Content moderation has evolved significantly with advancements in machine learning, particularly through the adoption of LLMs. Traditional approaches to hate speech detection, such as logistic regression, Support Vector Machines (SVMs), and deep learning architectures [1], achieved reasonable accuracy on basic classification tasks but faced challenges with sarcasm, indirect abusive language, or reclaimed slurs, limiting their effectiveness [2].

Recent research has highlighted the potential of LLMs, such as BERT, which overcome these challenges by leveraging Transformer architectures to capture deeper linguistic relationships [3]. For instance, Twitter's T-BERT, a domain-specific adaptation of BERT, has demonstrated substantial improvements in identifying harmful UGC by adapting to the unique linguistic landscape of social media [4]. While LLMs have shown promise, challenges remain. For example, Springer (2022) discusses difficulties in detecting subtle toxic content in social media contexts [5]. This highlights the need for continued refinement of LLMs to better balance accuracy with fairness in moderation tasks.

Building on these findings, our project aims to utilize LLMs for moderating multimodal UGC. Unlike existing systems that focus solely on classification, our approach includes suggesting constructive revisions to align posts with CG.

4. Data and Data Processing

Our project leverages the MMHS150K (Multimodal Hate Speech) dataset, which contains 150,000 tweets. We additionally built a web scraper to handle real X (formerly known as Twitter) data. To ensure consistency, the web scraper coerces data into the same format as the MMHS150K dataset. These datasets comprise textual and visual components. Images were resized to ensure uniform input dimensions for our models and referenced by tweet IDs. Labels were processed in accordance to each dataset as shown below.

4.1. MMHS150K Dataset

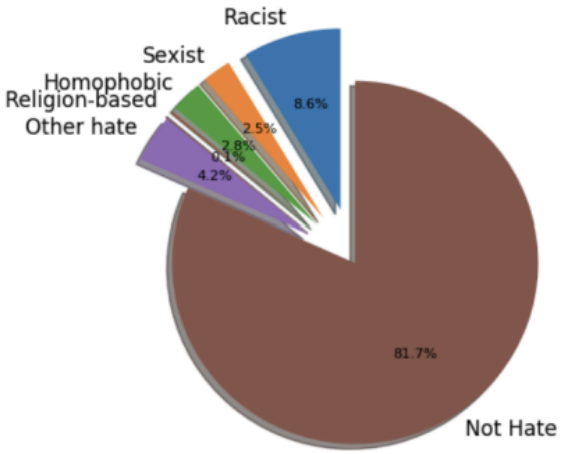


Figure 2: The MMHS150K tweet classifications

The MMHS150K dataset differentiates between multiple hate categories (Figure 2), which are annotated by three reviewers. The MMHS150K dataset exhibits a class imbalance: Hate (18.3%) vs. Not Hate (81.7%). We flattened all hate categories into a single binary classification, Hate (1) and Not Hate (0), where all hate-related labels (e.g., racist, sexist) were grouped into a single hate category. Then, we used a majority voting mechanism (i.e. arithmetic mean) to dynamically aggregate the flattened labels from multiple annotators; the final label of the datapoint was Hate (1) if the arithmetic mean was greater than 0.5, and Not Hate (0) otherwise (Figure 3). This ensured a more robust labeling process by consolidating annotator opinions and reducing individual biases.

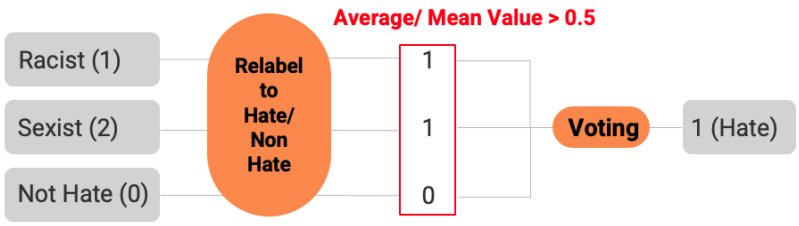


Figure 3: Diagram of flattening labels and the majority voting mechanism

An example from the MMHS150K dataset is shown in Figures 4 and 5.


```
"1024386871542145026": {
  "tweet_url": "https://twitter.com/user/status/1024386871542145026",
  "labels": [
    0,
    0,
    0
  ],
  "img_url": "http://pbs.twimg.com/ext_tw_video_thumb/1024386742072168448/pu/img/V2_V7OtvARNUOTxI.jpg",
  "tweet_text": "Rube Golberg success!! #D93Innovates #summertech2018 https://t.co/vuGh6bybVc",
  "labels_str": [
    "NotHate",
    "NotHate",
    "NotHate"
  ]
},
```


Figure 4: MMHS150k dataset sample marked as hate-speech

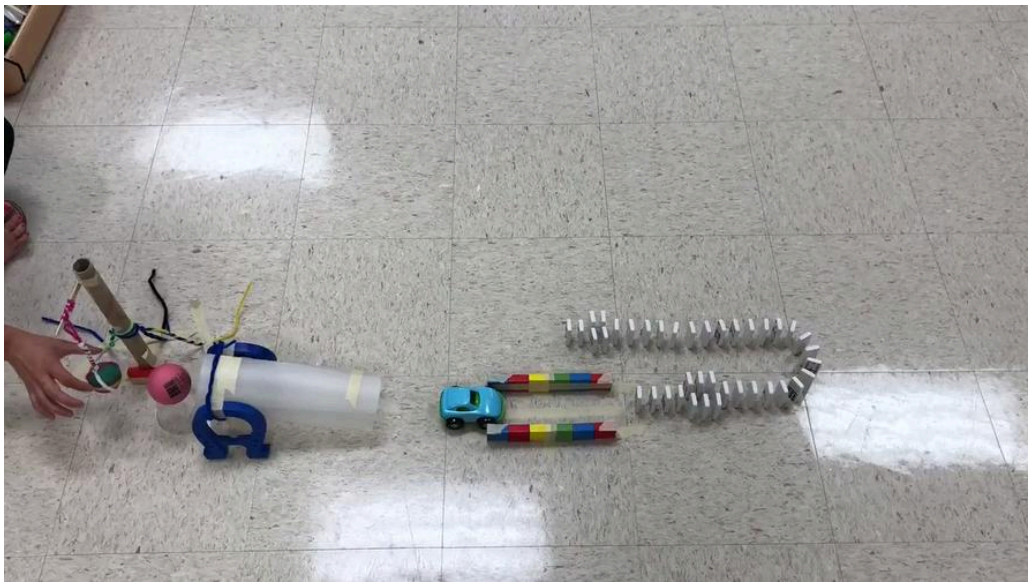


Figure 5: The associated image to the data sample in Figure 4

4.2. MMHS150KCuratedSmall Dataset

To address the class imbalance, we curated a subset of 26 examples from the original dataset. This new dataset, MMHS150KCuratedSmall, features a balanced class distribution: Hate (50%) and Not Hate (50%). Furthermore, we manually verified and split the flattened labels from §4.1 into image_only_labels and text_only_labels. These labels determine whether each respective constituent contains hateful content. An example from the MMHS150KCuratedSmall is shown in Figure 6.


```
"1024398008899780609": {
  "tweet_url": "https://twitter.com/user/status/1024398008899780609",
  "labels": [
    0,
    2,
    5
  ],
  "img_url": "http://pbs.twimg.com/media/DjdkgbVAAA0dqI.jpg",
  "tweet_text": "who's this c**t claiming to be jughead jones https://t.co/MD4MSdSlvy",
  "labels_str": [
    "NotHate",
    "Sexist",
    "OtherHate"
  ],
  "image_only_label": [
    0
  ],
  "text_only_label": [
    1
  ]
},
```


Figure 6: Manually labeled data for MMHS150KCuratedSmall Dataset

4.3. Web-Scraped Dataset

To enrich our evaluation, we manually collected 20 real-world examples from Twitter/X using a custom web scraper. These examples were manually labeled by our team and formatted to match the structure of MMHS150KCuratedSmall. Similar to the MMHS150KCuratedSmall dataset, we also generated the image_only_labels and text_only_labels. This Web-Scraped dataset is imbalanced, with a distribution of Hate (25%) and Not Hate (75%). Since we collected this from the most recent Tweets without cherry-picking, we believe this reflects a more realistic distribution. More examples can be seen in §12.1.

5. Architecture and Software

The system consists of three primary components: a web scraper, the Moderator models, and the Editor model. Each component interacts seamlessly to analyze, process, and provide actionable feedback to users (Figure 1). The exact prompts used in each model can be found in the GitHub repository §12.2.

5.1. Web Scraper and Data Preprocessing

The ingestion stream scrapes posts from Twitter/X and formats them into a JSON structure keyed by tweet_id. Metadata and textual content are stored as values, while image data is saved separately and referenced using the same tweet_id. This format is identical to the MMHS150K dataset.

5.2. Moderator Models

To handle the multimodal nature of tweets, we separated them into text-only and image-only components. Each component is separately handled by a GPT model to classify whether the respective component is compliant with CG.

5.2.1. Textual Moderator

A GPT-4 model analyses textual content against CG. It identifies whether the text is compliant and lists specific violations, if any, and provides detailed explanations.

5.2.2. Visual Questioning and Answering (VQA) Moderator

The visual content is analysed by the GPT-4o VQA model, which evaluates images based on their "image-only" compliance with CG. This step ensures that violations in images are identified independently of their associated textual context.

5.3. Editor Model

If the Textual Moderator detects violations, the content is passed to the Editor GPT-4 model, which generates revised text that aligns with the CG. Rather than simply replacing offensive words with synonyms, the Editor may change sentence structure and content, even rewriting the entire Tweet if necessary, while retaining as much of the original message's intent as possible. The revised content and explanations of the changes are then provided to the user.

5.4. Output Aggregation

The outputs of both Moderator models (textual and visual) and the Editor model are consolidated into a comprehensive response. This response includes:

- Compliance assessment for each Moderator Model
- Detailed explanations of violations.
- Revised content, if applicable.

6. Baseline Model or Comparison

The Post++ system as a whole was evaluated on two fronts. We determined compliance classification success based on each Moderator model's output evaluated against the image_only_labels and text_only_labels respectively.

We also determined the revisioning effectiveness of the Editor by running the revised textual content back into the Textual Moderator to determine whether the Editor's changes made a difference. This is shown in the architectural diagram (Figure 1) by the dotted-dashed lines. Notably, this evaluation style primarily depends on the recall of the Textual Moderator (i.e. how well the Textual Moderator catches all hateful content as non-compliant).

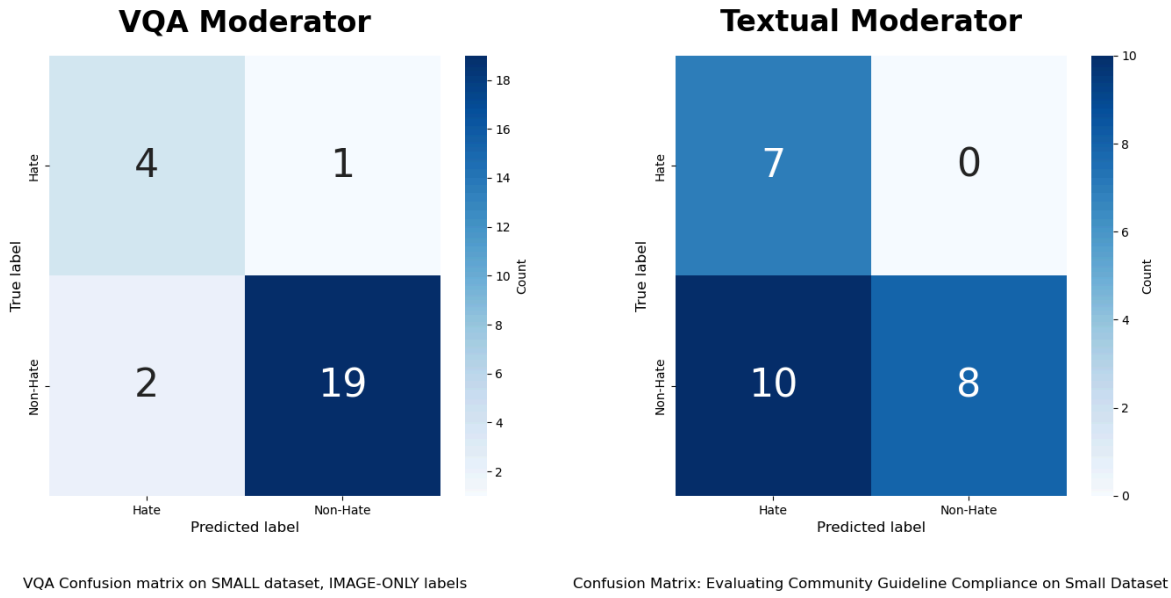
7. Quantitative Results

The performance of each model is shown in Table 1.

Model	Accuracy	Precision	Recall	F1-Score
VQA Moderator	88.5%	66.7%	80%	72.73%
Textual Moderator	60%	41.18%	100%	58.33%
Text Editor	94.12%			

Table 1: Evaluation metrics across all Models

In moderation systems, while minimizing false hits is ideal, it is more important to identify and flag non-compliant UGC. False positives only require unnecessary Editor revisioning calls whereas false negatives may erroneously approve non-compliant UGC. As such, this system is primarily concerned with recall. We can see high recall in both of our Moderator models. This shows that our system is performant for identifying hateful content.



VQA Confusion matrix on SMALL dataset, IMAGE-ONLY labels

Confusion Matrix: Evaluating Community Guideline Compliance on Small Dataset

Figure 7: Confusion matrices for each Moderator model on the MMHS150KCuratedSmall Dataset.

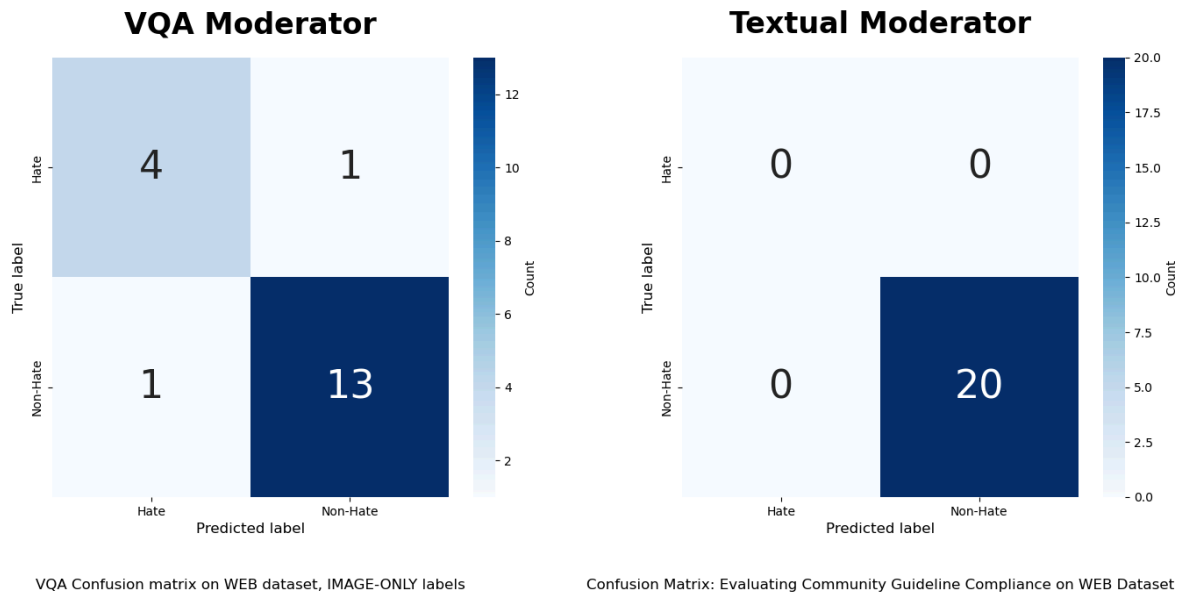
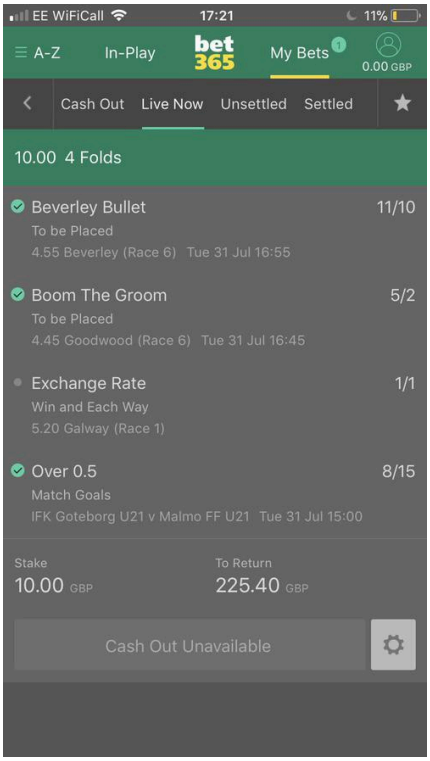


Figure 8: Confusion matrices for each Moderator model on the Web-Scraped Dataset.

As mentioned in §6, the Editor's performance was evaluated by running the revision outputs through Textual Moderator to determine whether the revised text was now compliant with CG. Our model achieved around 94% success in that regard.

8. Qualitative Results

Table 2 is an example from the MMHS150KCuratedSmall dataset with textual hate. This example illustrates the multimodal classification process and textual revisioning. Numerous derogatory and offensive terms have been censored in this document.

<table>
<tr><th>Input</th><th>Output</th></tr>
<tr>
<td>


"Ruby Walsh is a cheating f**k man, bailed on the first. Proper mutant c**t man #exchangerate https://t.co/j3S4i2FD1S"
</td>
<td>
<pre>
"1024329522869616640": {

 "Compliance": false,

 "VQA Response": "This image ... does not seem to contain any violent content, ...",

 "Moderator Reasoning": "",
 "Violations": [
   "Abuse/Harassment",
   "Hateful Conduct"
 ],

 "Revised Text": "Ruby Walsh didn't deliver in the first race. Very disappointing performance. #exchangerate https://t.co/j3S4i2FD1S",
 "Revision Explanation": "The original text contained offensive language ..."

},
</pre>
</td>
</tr>
</table>

Table 2: Example of Textual Hate Content in UGC

This example highlights the Editor's ability to perform nuanced revisions, that goes beyond simply substituting problematic words. It effectively preserves the original intent of the Tweet while accurately inferring context, such as identifying a connection between "Ruby Walsh", horse racing, and this betting context.

Table 3 represents another example, this time with multimodal hate content.

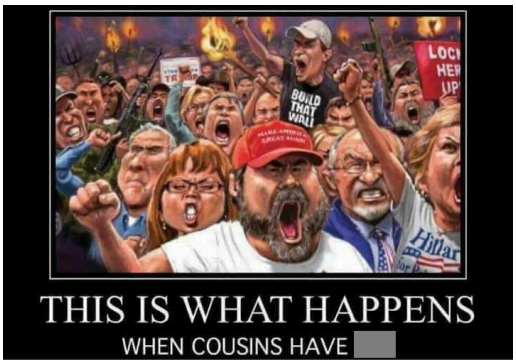
Input	Output
 "@RawStory I got your r*****k rebellion right here... https://t.co/ud8qtKEOJG"	"1024603824051421186": { "Compliance": false, "VQA Response": "The image does not comply ... contains hateful conduct ... of political affiliation or perceived socio-cultural background...", "Moderator Reasoning": "The term \"r*****k\" is used in a provocative and derogatory manner ...", "Violations": ["Hateful Conduct"], "Revised Text": "@RawStory I've got a perspective on the rebellion right here... https://t.co/ud8qtKEOJG", "Revision Explanation": "The change involves removing the provocative term ..." },

Table 3: Example of Multimodal Hate Content in UGC

This shows both Moderators flagging this multimodal hate content as non-compliant and the Editor revising the Tweet text to remove derogatory connotations.

Conversely, Table 4 shows an example where the Editor missed a derogatory term in UGC.

Input	Output
 "@SailorP0ON @SAVAGEYAYO t**p is ok, but t****y carries negative connations, like \"n****r\" https://t.co/sSXa12HgF8"	"1024393263526998016": { "Compliance": false, "VQA Response": "This image contains ... racial slurs ...", "Moderator Reasoning": "This text uses derogatory ... slurs against two groups of people: the transgender community and Black people...", "Violations": ["Hateful Conduct"], "Revised Text": "@SailorP0ON @SAVAGEYAYO t**p is ok, but we need to respect everyone's identity https://t.co/sSXa12HgF8", "Revision Explanation": "... removed the derogatory terms originally used in the text that were offensive to both the transgender and Black communities..." },

Table 4: Example of Missed Revision for Textual Hate in UGC

9. Discussion and Learnings

9.1. Performance

From §7, we can see that both Moderators have high recall. This shows that the Post++ system is performant at flagging hateful UGC. However, the Textual Moderator struggles with mediocre precision.

After sampling several false positives, it is apparent that the Textual Moderator model is sensitive to words used in insults and is biased towards flagging many insults as hateful. This is especially noticeable for regional insults where the majority of annotators labeled this as Not Hate while the Textual Moderator flagged it as Hate (Figure 9).

"@The Trump Train @realDonaldTrump Read this, you thick twat"

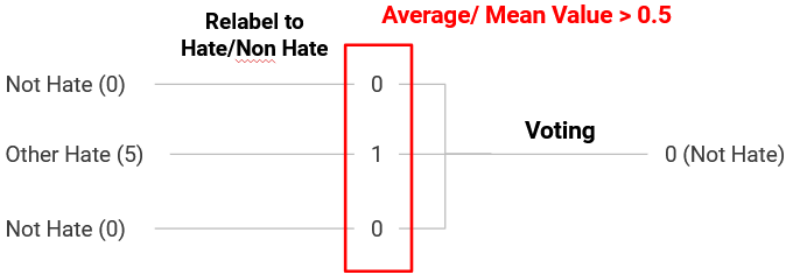


Figure 9: Textual Moderator False Positive sample

As shown in §7, our Editor achieved about 94% effectiveness in revising hateful UGC. Combined with the high recall in our Moderators, this shows that the Post++ system is performant in UGC moderation and revisioning.

9.2. Limitations

While our Moderators were effective, we encountered situations where the Moderators would reject certain inputs as seen in Table 2, Textual Moderator. This is likely because OpenAI restrains their models to not engage with derogatory, textual behaviour or identification/analysis of specific individuals (VQA). However, moderation is specifically concerned with these goals. Our system tolerates the failure of either Moderator and will still attempt to determine UG compliance.

9.3. Improvements

Experimenting with other models could lead to cost savings and improved performance. Currently, our system costs around 1.60 CAD for a batch of around 20. Replacing the Moderators with GPT-3.5 models will cut down on around 95% of the cost, although whether it remains performant has not been determined. Additionally, using a GPT-o1 model for the Editor may lead to fewer rejections and a deeper analysis of textual material.

10. Individual Contributions

David	Chao-Lin	Olivia	Yi-Fan
Unified data processing pipeline Curated the MMHS150KCuratedSmall dataset + Manual Labels VQA prototyping with LLaVA model Set up GPT-4o VQA API calls and prompts Evaluate performance of the VQA Moderator Managed GitHub Project and helped others with fixing GitHub troubles Worked on the reports	Implemented X web scraper Manually annotated the Web-Scraped dataset Refactored the main.py script into modular components Wrote README.md explaining project setup and functionality Worked on the reports	Set up Community Guideline & Prompt Engineering for Textual Moderator Textual Moderator Prompt Engineering Manual labeled the Web-Scraped dataset Fine-Tune Moderator model Worked on the reports	Set up API calls for Textual Moderator and Editor Textual Moderator and Editor Prompt Engineering Manual labeled the Web-Scraped dataset Evaluate the performance of the Textual Moderator and Editor Worked on the reports

11. References

[1] D. Kumar, R. Mamidi, and M. Srinivas, "Hate Speech Detection Using Machine Learning and Deep Learning Techniques," Springer, 2022, sec. 4.1.

[2] P. Fortuna and S. Nunes, "Hate Speech Detection: Challenges and Solutions," PLOS ONE, vol. 14, no. 8, 2019.

[3] J. Cai et al., "Watch Your Language: Investigating Content Moderation with Large Language Models," arXiv preprint arXiv:2309.14517, 2023.

[4] J. Li, S. Ji, T. Du, B. Li, and T. Wang, "Analyzing the Use of Large Language Models for Content Moderation with ChatGPT Examples," ACM Digital Library, 2023.

[5] V. U. Gongane, M. V. Munot, and A. D. Anuse, “Detection and moderation of detrimental content on social media platforms: current status and future directions,” Social Network Analysis and Mining, vol. 12, no. 1, Sep. 2022.

12. Appendix

12.1. Manual labeling in Web Scraped Data


```
"1783145468052054092": {
  "img_url": "https://pbs.twimg.com/media/GL6fQWPaMAAeIFY?format=jpg&name=medium",
  "labels": [
    0, 0, 0, 0
  ],
  "tweet_url": "https://x.com/unknown_th82411/status/1783145468052054092",
  "tweet_text": "Just half?",
  "labels_str": [
    "NotHate", "NotHate", "NotHate", "NotHate"
  ],
  "image_only_label": [
    0
  ],
  "text_only_label": [
    0
  ]
},
```

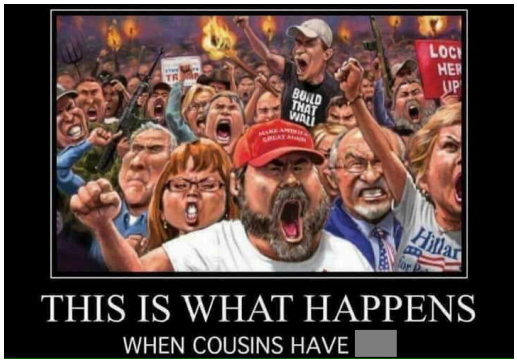

Figure 10: Manually labeled data for web-scraped UGC

Figure 11: The associated image to the data sample in Figure 10

12.2. GitHub Repository Link

GitHub repository: https://github.com/ece1786-2024/Post

12.3. Full Input/Output to Post++

<table>
<tr><th>Input</th><th colspan="2">Output</th></tr>
<tr><td rowspan="3"> 

"@RawStory I got your r*****k rebellion right here... https://t.co/ud8qtKEOJG"</td><td>VQA Moderator</td><td>"VQA Response": "The image does not comply with the Platform Content Guidelines because it contains hateful conduct by attacking people on a presumed basis of political affiliation or perceived socio-cultural background. The image caricatures and mocks a group of people with derogatory language.\n\n0",</td></tr>
<tr><td>Textual Moderator</td><td>{
"compliant": false,
"violations": [
"Hateful Conduct"
],
"explanations": "The term \"r*****k\" is used in a provocative and derogatory manner presumably attacking a group of people based on their rural background or sociocultural identity. This falls under Twitter's Hateful Conduct guideline which disallows the targeting of individuals on the basis of their group identifiers. However, the link provided is not evaluated in this assessment. If the link leads to content that violates other guidelines, those would need to be addressed separately.",
}</td></tr>
<tr><td>Textual Editor</td><td>{
"revised_text": "@RawStory I've got a perspective on the rebellion right here... https://t.co/ud8qtKEOJG",
"explanation": "The change involves removing the provocative term \"r*****k\", which was used in a derogatory manner, and replacing it with a neutral phrase \"a perspective\". This ensures the revised text does not target or stereotype any group, respecting Twitter's Hateful Conduct guideline which prohibits targeting individuals based on group identifiers."
}</td></tr>
</table>

Table 5: The full input and output of the Post++ system for the example shown in Table 3