CS489 Final Report

Cleanity: The Social Harmony

20180532 Heechan Lee
20200433 Gyuho Lee
20210277 Hojun Park

GitHub repository: https://github.com/Gabul99/cs489-project/

Introduction

In modern society, the role of online communities is extremely powerful. They serve as club-like spaces where people with shared interests can gather to exchange information and communicate with each other, while also functioning as platforms for debating diverse opinions. A notable example of this is the declaration of martial law on December 3rd. On Naver News alone, there were 920,000 comments on December 4th, and 960,000 comments during the 1st impeachment vote on December 7th [4].

During emergencies, obtaining information through the Internet has become second nature for many. When martial law was declared, the sudden surge in traffic temporarily disrupted some of Naver's services [2, 3]. According to one survey, 73.6% of the public first learned about martial law through social media, and 96% believed that social media played a significant role in its termination [1]. Experts also point out that it is practically impossible to completely shut down the internet and communication networks in modern society [3, 5], which again shows the strong power of online communities.

As online communities continue to evolve, they have become embedded in modern culture. For instance, one of South Korea's largest online communities records an average of 2.9 million daily visitors, and it has even been the subject of academic research [7, 8].

However, the development of online communities is a double-edged sword. The biggest issue is that it has become too easy for anyone to express their opinions. For instance, while comment sections may seem to reflect a wide range of public

opinions in general, they are often dominated by a radical minority [9]. Another significant issue is *trolling*. Trolling is a global problem faced by all online communities, where some users purposely harass others, dispiriting the healthy growth of these platforms [10].

Attempts to address these problems, such as restricting comment features, have also led to unintended conclusions. For example, Daum replaced its comment section with the “Time Talk” feature but ended up losing many users [6]. This demonstrates that wildly limiting communication between users can reduce overall participation in the community. This leads modern communities to have their own management systems that automatically catch inappropriate behaviors.

Existing Problem

Currently, most community management systems rely on rigid, rule-based approaches to moderation. This rule-based method has a high chance of making faulty decisions. A well-known example is the ‘Scunthorpe Problem’, which is due to the filter being so aggressive that it blocks the word that isn’t problematic at all [11]. Making these systems more flexible obligatorily requires the involvement of human moderators, which increases the burden on resources and time. We were able to find out that moderators have to spend hours managing the community, as well as going through online harassment or unintended exposure to harmful content [12].

Another problem is that users frequently lack awareness or understanding of the platform's rules, leading to unintentional violations and ineffective compliance. All communities have rules, and users do not read them even though the rules vary from community to community. It wasn’t directly related to the community rules, but we were able to find out that site users only read around 100 words, and less than 20% of the total words on average [13]. They just scan through the information they need or do the thing they want without any consciousness of the rules.

Due to these existing problems, we suggested making a new community moderation system, Cleanity, with a novel approach that can solve the strict evaluation problem from existing solutions and ultimately contribute to resolving the malicious troll

issues, pressure of moderators, and user's perception about community rules.

Solution

1. Custom Rules

To address the issue of strict rule application, we propose the **Custom Rules** feature. Even within communities centered around the same topic, each community may have a distinct character and require different rules. Allowing community administrators or members to create and apply rules tailored to their specific needs through mutual agreement would be more effective. The Custom Rules feature in Cleanity facilitates flexible rule application that suits each community.

The pipeline enabling this Custom Rules feature is built upon a Large Language Model (LLM) service. Unlike rule-based censorship systems that strictly filter predefined words, utilizing LLMs allows for a more flexible understanding of the intent behind the rules, user behavior, and the overall context of the community.

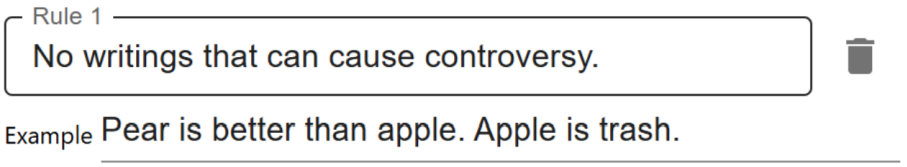

Figure 1. Example of a set of Custom Rules and example that violates it.

We enhanced the pipeline through iterative testing of the Custom Rules feature on various types of comments. First, we make users write examples that violate each rule. Since LLM's judgment based solely on rules can work differently from the moderator's original intention, we provide examples to ensure that it works as the moderator intended. Inspired by few-shot prompting, commonly used in prompt engineering, this approach was effective in conveying the intent of Custom Rules, given the diverse situations across communities.

Additionally, we recognized the importance of making the tool user-friendly from the perspective of community moderators. After system iterations, we found that the most important factor in the performance of this system was the quality of the rules. Non-specific rules, such as "don't be mean to others," reduced the value of the system. However, it was challenging to create a set that was specific enough and still

aligned with the moderator's intentions. Since writing and refining rules can be challenging, we provide a Rule Refinement feature powered by LLMs. In the refinement feature, LLM reviews drafts of user-drafted rules and revises them to make them more specific.[1] We also offer editable presets to help moderators easily understand the structure of the rules.

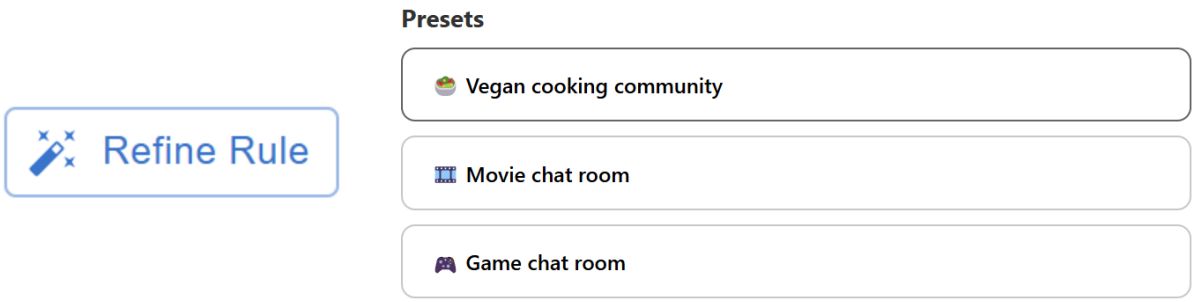

Figure 2. Rule Refiner and Example of preset rules that we implemented.

The rules can be tested in the Playground area. The Playground feature allows users to verify whether specific comments are filtered based on the current rules. This functionality helps moderators evaluate whether their set of rules effectively filters inappropriate content or performs correctly in reverse scenarios. Through this iterative process, moderators can experience building and refining appropriate rules.

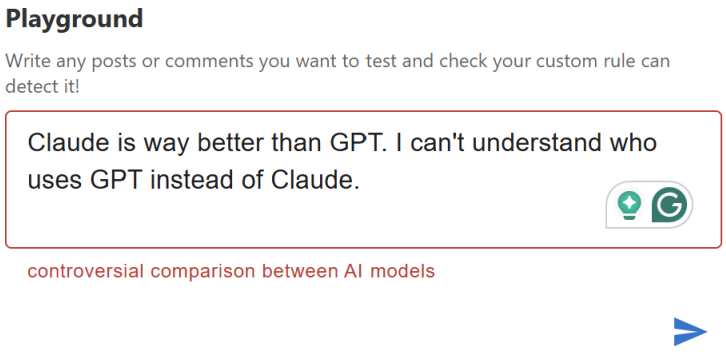

Figure 3. Playground that shows how rule works.

2. Prompt Feedback

As previously mentioned, *rule awareness* among community members is essential. There are two main approaches to fostering *rule awareness*. The first approach is to force users to read the rules. However, considering that people often skip reading terms and conditions in financial situations involving their money, it is nearly

[1] For the prompt used to refine the rule, see RULE_REFINE_PROMPT in the link below. https://github.com/Gabul99/cs489-project/blob/main/cs489-project-front/src/prompts.ts

impossible to force users to thoroughly read and understand community rules. The second approach is to enable users to naturally learn the rules through their activities or by observing others' actions (e.g., posting comments, uploading chats, etc.).

We propose an approach that provides **prompt feedback**, allowing users to naturally become aware of the rules. If a user attempts an action that might violate the current rules, the system informs them of which rule they are violating and explains why it considers the action a violation.

This feature operates based on the final response of a Custom Rules pipeline powered by an LLM. When a user submits a comment on the demo page, the system evaluates it against the Custom Rules using an LLM-as-a-Judge.[2] If a violation is detected, the user receives a red hint text explaining why their comment violates the rules.

This approach offers several advantages; It reduces the time inappropriate content is visible to users without requiring immediate intervention from a moderator. This alleviates the workload of moderators. By automatically giving feedback about inappropriate user actions and gradually increasing *rule awareness*, the system can significantly reduce the burden on volunteer moderators, who often manage communities voluntarily. As more users become familiar with the rules, conflicts among users and between users and moderators can be minimized. This increased awareness encourages community members to actively collaborate in refining and developing rules, contributing to the creation of a better community overall.

3. Alternative Suggestion

After receiving feedback, users can choose to either edit and repost their comments or abandon their posts altogether. However, editing and reposting their comments to avoid breaking the rules requires a lot of effort from users. It's quite a lot of work for users to come up with comments that are different from what they originally thought and retype them. Our goal is to avoid creating too many obstacles for users in writing their comments. Therefore, we introduced the **Alternative Suggestion** feature to

[2] For the prompt used to refine the rule, see RULE_EVALUATE_SUGGEST_PROMPT in the link below.
https://github.com/Gabul99/cs489-project/blob/main/cs489-project-front/src/prompts.ts

help users easily adopt and utilize suggested comments.

If the feedback confirms that the user's action violates a rule, Cleanity leverages an LLM to suggest one or two alternative comments that retain the original nuance as much as possible while working with the rules. Users can select one of these suggestions and freely edit it further. Even if they choose not to use the suggestions, these alternatives can serve as guidelines for the user.

On our demo page, we provide a simple chat simulation environment to demonstrate this feature. If a user writes a message that violates a rule, they can choose to send a modified version based on the suggestions or click the ‘Send Anyway’ button to post the original message. This approach can make users feel free to ‘control’ their actions.

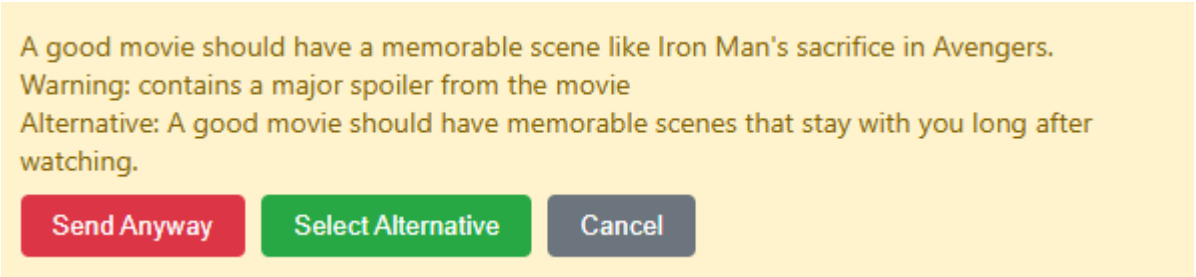

Figure 4. Example of chat suggestion after detecting a breach of rules.

For example, if a user leaves a message saying "A good movie should have a memorable scene like Iron Man's sacrifice in Avengers." in a movie community where spoilers are prohibited, a warning window like the one above will pop up. Users can check the warning window and remind themselves that the message may be a spoiler to some people and maybe break the rules. They can click the Select Alternative button to leave a comment without having to delete, think, and rewrite the message.

When applied to a real community system, this feature can allow users to retain control over their actions while enabling moderators to detect and review potential issues more efficiently. Depending on the community, rather than restricting freedom of expression, this method offers appropriate interventions that enhance *rule awareness* and iterate action correction for users. At the same time, it supports moderators by providing a tool for efficient community management. This balance

ensures that the system respects user freedom while maintaining order and reinforcing community rules.

Conclusion

Implications

Our service Cleanity has revolutionized community management by enabling the creation of more flexible and diverse rules through Custom Rules, while helping even beginners build high-quality rule sets with the rule preset and rule refine functions. The platform offers automatic Prompt Feedback, which eliminates the need for moderator intervention, provides immediate regulation of rule-violating posts before they can cause harm, and informs users of their mistakes to encourage behavioral correction. Additionally, the Select Alternative button streamlines the process of editing comments on posts that violate rules, significantly simplifying the traditionally laborious task of comment moderation.

Limitations

High LLM dependency

Our service heavily relies on responses from LLM. Checking for rule violations in messages and suggesting alternatives for messages that violate rules are based on LLM responses. Therefore, if LLM makes a judgment different from the intention of the community manager, it is greatly affected. The LLM we confirmed during the development process can be largely divided into two types. The first is when the rule is poorly set without specificity, and the second is when LLM itself malfunctions. To respond to both cases, we first tried to prevent poor-quality rules from being set by adding Rule preset and Rule refine functions. We also tried minimizing the impact when LLM malfunctions by allowing users to "Send Anyway" even when rule violations are detected.

Delay due to LLM response

There were a few seconds of delay between when a user sent a message and when they received feedback, which lowered the quality of user experience. This is because the process of receiving a response from LLM takes a long time in our operation processes. In particular, we conducted user tests assuming a chat situation in various online community situations, and in the chat situation, the few seconds waiting for a response from LLM felt more critical. In situations where a quick response is less necessary, such as posting a relatively long post, the limitations of our system may be less noticeable. However, in environments that require quick responses, such as chat, the limitations are expected to be noticeable.

Future Work

Improving high LLM dependency through moderator intervention

Due to high LLM dependency, there are cases where messages intended to be regulated are not regulated (False Negative), and messages intended not to be regulated are not regulated (False Positive). To improve this, it is necessary to develop a method for community moderators to intervene in the rule system.

For the case of False Negative, a function is needed for moderators to review chat history and regulate it afterward. Messages regulated afterward by moderators can be added as shots to LLM, thereby improving the accuracy of future operations.

In the case of False Positives, a user reporting function is required. When a user is detected to have a rule violation in a message even though the user did not violate the rule, they can request a review by reporting it to the moderator. When the requested message judged by the moderator doesn’t seem to violate the rule, it can be added as a shot to LLM as a case of non-violation of the rule. The regulation accuracy of LLM can be improved with the added shot.

In addition, having LLM review the community's posts and understand the context can reduce LLM's incorrect judgments. If LLM determines that a post contains information that is necessary for determining whether a rule is violated, it extracts that information and uses it as part of the prompt. For example, let's say

there is a sports team fan community where cheering for players from other teams is prohibited. In this situation, if a player from another team transfers and joins the team, LLM cannot help but be confused about whether cheering for him is prohibited or not. To reinforce the accuracy of judgment, LLM can save the post that a certain player has joined and use it for future judgments.

Improving LLM response delay

Currently, the LLM response time is getting longer due to long explanations and many shots being inserted as prompts in the LLM. There are various ways to reduce the LLM response time, but the most effective way is to simplify the prompt. It is necessary to remove unnecessary content from long explanations through testing. In addition, the shots should be simplified by studying how the LLM learns efficiently while inserting fewer shots.

Replacing the model in use with another LLM can also be considered. First, replacing the LLM with a mini model with fewer parameters can help with the response time. Also running it locally rather than via API can help with response times.

Multi-modality support

In the community, rule violations are not only text but also images/videos. Obnoxious images, images out of the blue that don't have any relations with the community hinder communication. Therefore, adding a method to analyze visual data and determine whether they violate the rules will enable a better user experience. With prompts about the rules, we can use VLM (Vision-Language Model) such as OpenAI CLIP and Google Cloud Vision API to determine whether the image complies with the rules.

References

[1] 유지희. (2024, December 8). SNS로 계엄령 선포 처음 알았다. . .카톡 감시 당할라 공포. 한국경제. https://www.hankyung.com/article/202412068632g
[2] 손엄지지. (2024, December 3). 네이버 뉴스 댓글, 20여분 만에 정상화. . ."트래픽

급증"(종합). 뉴스*1*. https://www.news1.kr/it-science/general-it/5619755

[3] 선담은. (2024, December 4). 포털 카페 '먹통'에 놀란 가슴. . .계엄 때 '인터넷 완전 차단' 가능할까. 한겨레. https://www.hani.co.kr/arti/economy/economy_general/1170944.html

[4] 네이버 데이터랩 : 댓글통계. (n.d.). https://datalab.naver.com/commentStat/news.naver

[5] 이가람. (2024, December 12). 계엄령에 휴대폰 안 되면 어쩌나. . .이통사 답변은 "사실상 통제 불가능." 매일경제. https://www.mk.co.kr/news/business/11186625

[6] 댓글 없앴더니 이용자 뚝. . .포털 다음의 '손해 본 선택.' (2023, August 31). 세상을 바꾸는 시민언론 민들레. https://www.mindlenews.com/news/articleView.html?idxno=4945

[7] *2022* 대한민국 커뮤니티 보고서. . . 모든 것이 이곳에서 시작된다. (2022, September 5). 주간조선. https://weekly.chosun.com/news/articleView.html?idxno=21824

[8] 박인성. (2022). 밈과 신조어로 읽는 인터넷 커뮤니티의 부족주의—남초 커뮤니티의 정서적 평등주의와 위임된 성장서사. 대중서사연구, 28(2), 59-93.

[9] 박원경. (2021, January 5). [사실은] '0.03%가 30% 차지'…포털 뉴스 댓글은 여론인가? *SBS NEWS*. https://news.sbs.co.kr/news/endPage.do?news_id=N1006153616

[10] Delaney, J. (2016, July 14). *Opinion: Online trolls are ruining social-media marketing: Online-media companies and social-media platforms are closing comments sections*. MarketWatch. https://www.marketwatch.com/story/online-trolls-are-ruining-social-media-marketing-2016-07-13

[11] Tom Scott. (2016, June 6). *Why web filters don't work: Penistone and the Scunthorpe problem* [Video]. YouTube. https://www.youtube.com/watch?v=CcZdwX4noCE

[12] *What are some of the hardest moments you had as a moderator?* (2019, May 14). Discourse Meta. https://meta.discourse.org/t/what-are-some-of-the-hardest-moments-you-had-as-a-moderator/117677

[13] Nielsen, J. (2024, February 2). *How little do users read?* Nielsen Norman Group. https://www.nngroup.com/articles/how-little-do-users-read/