Aidan Evans

Professor Feigenbaum

CPSC 610: Topics in Computer Science and Law

14 May 2021

Automatic Content Moderation in Real-Time Communities

1. Introduction

In this study, I aim to explore how automatic content moderation is conducted on the online communication platform Discord. Prior research has shown that automatic content moderation systems, such as Reddit's AutoModerator, greatly reduce the workload of online communities' human moderators (Jhaver et al., 2019). At the same time, while helping complete tedious, context-independent tasks, the introduction of automatic content moderation systems also imposes new work for human moderators, such as correcting false positives and editing the moderation system's settings to account for new trends (Jhaver et al., 2019). Unlike Reddit, Discord does not have an official automatic content moderation system that moderators can use to enforce community-specific rules; moderators on Discord rely on third-party applications that vary greatly on design and capability. Moreover, conversations on Discord happen differently than on Reddit: Discord provides a real-time chat environment which is better suited for "lengthy back-and-forth conversations" as opposed to the "threaded long-form conversations" which take place on Reddit (*451: Reddit-x-Discord*). While prior studies have used Discord to research the challenges of content moderation in voice-based communities (Jiang et al., 2019), the challenges moderation teams have when changing from Reddit to Discord (Kiene et al., 2019), and trends relating to how often moderation teams use bots (Kiene and Hill, 2020), no research to my

knowledge has been conducted to give an in depth analysis on Discord's use of automatic content moderation. Given the differences to other platforms, studying the use of automatic content moderation on Discord may introduce new insights into the collaboration between humans and technology online and introduce new issues in content moderation not encountered in previous studies.

2. Background

2.1 What is Discord?

Discord is a free online platform that gives communities a flexible way to communicate online. In December 2020, Discord reached 140 million monthly active users (Lunden, 2020). Any user can create a "server," sometimes called a "guild," which other users can join. A server can be either public or private; anyone can join a public server while users must be invited to join a private server. As of 2021, there are at least 13.5 million active servers per week (*About Discord*). Most servers tend to portray themselves as a community of individuals who share in some common interest, such as a specific game or project, while some exist as just an open community with no specific niche. Furthermore, each server is partitioned into "channels" where conversations between members of the server take place. Channels can either be text-based or voice-basted; text channels allow members to send messages and voice channels allow members to communicate using audio or video. The existence of multiple channels in one server allows for the organization of content into separate places; for example, most servers have a "general" channel for general, every-day conversations and then some other channel for posting funny images or for organizing events. Members can also have multiple roles which allow server moderators to assign users different permissions based on their roles; the association of permissions with roles causes the structure of Discord's moderation to reflect that of the

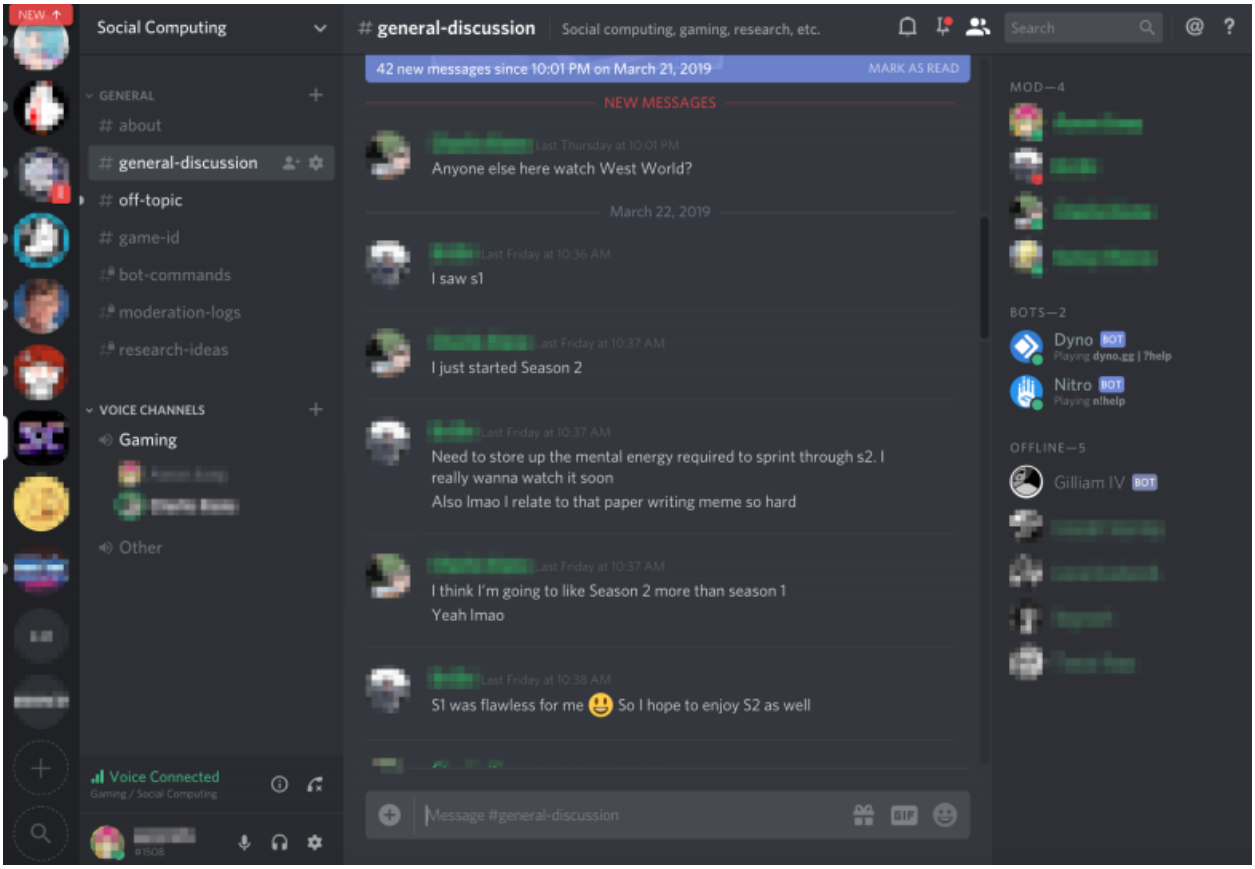

Figure 1: The standard layout on Discord. Messages are shown center. A list of the user's servers are shown on the very left as icons. To the right of the server list is the channels of the server the user is currently viewing. The pane on the very right displays the members of the server. (Image from Jiang et al, 2019)

permissions used to control access on UNIX file system (Zhang et al., 2020). Permissions can control what channels a member has access to and other server- or member-specific settings. The Discord application also lists the current members of a server on the right-hand-side of the screen, separated by member's roles and whether that member is currently online (see Figure 1).

2.2 How is Discord Moderated?

All users on Discord must abide by Discord's Terms of Service and Community Guidelines. Additionally, servers typically post their community-specific rules in a "rules" channel that only moderators can post in; this channel acts as a bulletin-board for additional rules

members of the server must follow. A server's moderation team then have the responsibility of enforcing Discord's Terms of Service, Community Guidelines, and the server's custom community-specific rules. Moderation teams are typically split between *administrators* and *moderators*. Administrators typically have full permissions on the server and moderators have the elevated permissions they need to carry out their job. Unless otherwise noted, this paper will henceforth use the term moderator to refer to any member of a server's moderation team.

Discord provides various tools to help servers moderate. I have already touched briefly on Discord's role-permission system. I explain some additional tools here as well.

2.2.1 Explicit Media Filter

Discord provides a central explicit media content filter which automatically scans uploaded media and checks if it contains explicit content. Moderators can turn on or off the explicit media content filtering in the server's settings or have special Not Safe For Work (NSFW) channels with the filter disabled just for those channels; moreover, the filter can be set to filter media uploaded by all member or just members without roles. Importantly, Discord scans all uploaded media – even if the explicit media content filter is turned off – for child sexual abuse imagery using the industry-standard PhotoDNA tool (*How we Investigate*).

2.2.2 Slow Mode

Slow Mode allows servers to limit the post frequency in specific channels. For example, servers can limit a channel to only one post per user every thirty seconds; this keeps users from spamming the chat and gives moderators more time to review posts before a conversation gets out of hand.

2.2.3 Verification Levels

Discord provides servers the ability to set up various verification levels that stop users who don't meet the criteria from joining the server. There are five verification levels possible.

- None: no restrictions, anyone can join
- Low: users must have a verified email on their Discord account
- Medium: users must be registered on Discord for longer than 5 minutes
- High: users must be a member of the server for longer than 10 minutes
- Highest: users must have a verified phone number on their Discord account

2.2.4 Third-Party Bots

Most automatic content moderation on a server, however, relies on adding third-party applications – called "bots" – to the server. Bots are used for a variety of things on Discord ranging from playing music in voice channels to helping moderators perform mundane actions. The providers of bots use Discord's API in order to interface with Discord's servers; moderators typically add a bot to their servers by clicking a bot-specific link that authorizes the bot to join. In regards to the configuration of moderation bots, moderators configure the bot by either typing bot-specific commands into their server's channel or if the provider of the bot has a graphical user interface hosted on their website, the user may login to the provider's website to configure the bot on some GUI, typically referred to as a dashboard.

Discord provides recommendations for moderators on what bots to use for automatic content moderation (*321: Auto Moderation on Discord*). All of these moderation bots provide some form of keyword detection and spam filtering but, overall, vary greatly in their abilities. For example, only two of the seven bots on Discord's auto moderation page provide filter configuration using regular expressions ("regex") – a tool that gives Reddit's AutoModerator

great flexibility in defining text filters (Jhaver et al., 2019). Some bots have intuitive user interfaces that allow for non-technical users to configure the bot, as opposed to a command-line text interface. In fact, only one of the seven bots Discord compares provides both regex filtering and a dashboard interface; this bot, however, lacks a robust anti-spam filter and raid detection[1].

2.3 Automated versus Automatic Procedures

To both avoid confusion and clearly define the scope of this paper, I will now briefly distinguish between two related terms which tend to be used interchangeably: automated versus automatic procedures. Automated procedures perform a sequence of operations to perform desired task. Automatic procedures, however, are performed without constant user interaction. Automatic procedures are a proper subset of automated procedures; in other words, every automatic procedure is also an automated procedure. For example, an anti-spam filter which removes spam from a chat is an automatic procedure because it searches for spam without the need of constant user interaction. A moderator using a bot to help remove the last ten posts via commands, on the other hand, would just be an automated procedure because the bot's action is a direct respond to the moderator's command and the bot will only perform this action when the moderator executes the command. Moderators use bots to carry out automated procedures – both automatic and non-automatic – that help aid moderation. While I may touch briefly on non-automatic procedures at certain points, this paper aims to identify how automatic procedures aid moderation in real-time environments.

3. Related Work

3.1 Automatic Content Moderation in Online Communities

[1] Raids will be discussed in section 5

Earlier work on automatic content moderation includes the use of automatic moderation on Reddit; specifically, Reddit's AutoModerator: a moderation tool available to subreddits to perform filtering of posts based on regular expressions. The Reddit AutoModerator allows moderators to filter suspicious links and offensive language, auto-approve posts based on users' account ages, and remove posts that are not in the correctly specified format (Jhaver et al., 2019).

3.2 Content Moderation on Discord

Current studies relating to content moderation on Discord do not focus on the use of bots for automatic moderation. Jiang et al. uses Discord to study on the challenges of content moderation in voice-based communities because of Discord's heavy use of voice-channels for communication (2019). Kiene et al. focuses on the challenges moderation teams have when changing from Reddit to Discord (2019). Finally, Kiene and Hill use Discord to study trends relating to how often moderation teams use bots; they concluded "that the effect of size was strongest for moderation tools that allowed the systematic logging and indexing of moderation actions like warnings and bans" (2020). In other words, the larger the community the greater need for automated moderation tools to help log actions.

3.3 Content Moderation in Real-Time Environments

Cai and Wohn investigate what moderation tools are available on the streaming platform Twitch. They conclude that "chat control involves real-time content management, and mods have to deal with information overload and to make decisions immediately, suggesting that mods in live streaming communities are undertaking a different type of time-sensitive psychological pressure than those in other communities" (2019). The chat environment on Twitch is fast-paced and conversations move quickly; because of this, moderators have little time to make decisions.

Like Reddit's AutoModerator, Twitch's also offer a filter-based approach to chat control by allowing moderators to automatically remove or flag messages for review based on keywords.

4. Method

In order to investigate, I interviewed 12 moderators of Discord servers about their experience using bots for content moderation. This study was approved by Yale University's IRB.

4.1 Recruitment

Subjects were recruited by contacting active moderators on public servers. Specifically, I joined popular public servers and if they had moderation bots, then I contacted the moderators of the server asking if anyone who had experience working with bots was interested. I also recruited participants by making an announcement on the Discord Moderator Discord (DMD), a private server run by Discord itself for trusted members of the moderation community.

4.2 Interviews and Participants

The study involved interviewing 12 Discord moderators from North America (6 moderators) and Europe (6 moderators). These interviews ranged from 33 to 135 minutes with an average interview time of 88 minutes. Interviews were conducted over Discord's voice chat, audio recorded, and then transcribed. Questions were asked in a semistructured interview split into two main parts: questions about their general experience moderating and questions about their experience interacting with bots for moderation purposes on Discord.

Participants' occupations included students, medical professionals, factory workers, software developers and more. Collectively, participants had moderated hundreds of servers

ranging from small servers to the largest on Discord – i.e., from <100 to ~800,000 members; server types included gaming, social, software development, special events, and more. Many participants also had experience moderating on other sites before Discord – including Reddit, Facebook Groups, Google+ Groups, Web or Gaming Forums, and IRC. Approximately one-third of participants were also currently or former members of the Discord Moderator Program, a program run by Discord itself to train moderators. Further details about age, gender, and experience are listed in Table 1.

Participant	**Country**	**Age**	**Gender**	**Years as a Moderator**
P1	Belgium	24	M	3
P2	Canada	21	M	5
P3	Netherlands	26	M	5
P4	Canada	24	M	~5
P5	Canada	26	F	11
P6	Finland	21	M	8
P7	France	23	M	6
P8	U.K.	30+	M	~20
P9	U.S.	25	Prefer not to say	~3-4
P10	Poland	28	M	7
P11	U.S.	24	F	7
P12	U.S.	31	F	~1

4.4 Data Analysis

Interview transcripts were analyzed using thematic coding as described by Braun and Clarke (Braun and Clarke, 2012). Transcripts were coded using a combination of both inductive and deductive thematic analysis, generating codes such as “bots important for logging” or “manual checking of bot actions.” Ultimately, after recursively reviewing codes, two primary themes unique to automatic content moderation in real-time environments were identified: (1) “heavy use of frequency-based algorithms” and (2) “use of multiple bots, despite possible redundancy.” I discuss these themes in sections 5.1 and 5.2, respectively.

5. Results

5.1 Frequency-based Algorithms

Unlike non-real-time environments, real-time environments have a higher demand on frequency-based algorithms. Frequency-based algorithms rely on keeping track of how often specific events occur. For example, keeping track of how many users have joined the server in the last then minutes or how many posts some user has posted in the last thirty seconds. The use of frequency-based algorithms are most prevalent in bots’ raid detection and anti-spam features.

Raids are events where a many users join a server within a short period of time with the purpose of causing havoc, e.g., spamming the text channels with many messages, spamming the voice channels with loud sounds, etc. While raids do happen with normal users joining, most raids use accounts controlled by bots – mirroring the effects of a Denial of Service attack. Some agent has many bot accounts joining and overloading the Discord server to the point where it can’t be used. P1 even reported that some raids use bots with verified phone numbers on their Discord accounts, circumventing the highest level of verification Discord provides:

Because if you put it on highest, that's the annoying part. And there are a lot and a lot of bot accounts… that have verified phone numbers. And if you pick the highest, it basically [doesn't work to prevent raids.]

The most useful way to prevent raids are to identify them before they occur and to do this, bots keep track of user join frequencies, i.e., how many new users have joined the server within a certain amount of time. P2 noted that the bot Gaius has an "automatic panic" mode that can be used to alert moderators when a raid may be occurring by keeping track of join frequencies:

For other communities, they'll utilize what's called automatic panic. So automatically, they will panic mode. If it detects say 10 people joining in the span of 21 seconds. They're automatically muted or notified in that your panic mode enabled because 10 people joined, here is their IDs…

P3 also noted that bots use frequency analysis in addition to other information in order to make a decision about raids:

From from what I know, is that it looks at account creation date, and patterns and user names and join dates. So when a lot of different users that have a new account join at the same second, they automatically get banned.

P5, P6, P7, P10, and P11 all noted similar things in regards to the use of frequency-based algorithms for raid detections/prevention.

Frequency-based algorithms also have an important role in spam detection systems. In regards to spam filtering, P2 mentioned how Gaius handles spam by keeping track of how messages a user has recently sent and sometimes the timeframe in which those messages were sent:

Most of it's fairly simple, just the basic is just like let's say five messages [sent] straight up... And then you have the more advanced level: in how many seconds?

Also, as P4 noted, moderation teams have to constantly update their systems in order to calibrate for what's a reasonable number of messages to send within a certain amount of time:

But as we got more members, we had to update you know, how many? How fast can people send messages?... We're always continuously updating…

As reported by Jiang et al., users may abuse voice-chats by leaving and rejoining voice channels because the action of leaving and joining a voice chat causes a sound to play for all members in that voice channel. P1 mentioned how the use of frequency-based algorithms help in moderating this specific issue in voice chats:

Voice channels are always a bit difficult. The only thing we can do with that as regards to both is check the traffic; if they're quickly joining and leaving… if they do it too much in the short time period, they're also auto-muted as well because then there's a high chance that either that they're just being stupid or being bored or just going through all the channels nonstop because they're bored or it's someone actively going through active voice chats and annoying people…

Moreover, because of the introduction of more frequency-based algorithms, more opportunities open up for false positives. Both P6 and P7 reported false positives arriving from lag. When users post messages but the messages do not get sent due to, e.g., a bad network connection, they are queued to be sent when the client application can reconnect; when this happens, all of the messages the user tried to send, get sent to the server at once, possibly triggering frequency-based anti-spam filters because *x* amount of messages were all posted in a

few milliseconds. As P6 noted in regards to a logging message he shared in regards to undoing the action of a bot:

And it says accidental spam, slow internet connection, because like that can happen. That happens in instances where like people's internet suddenly like cuts off for for a brief moment. So like when they're trying to send messages. So discord queues up those messages to send when the network connection comes back. And then if they like tried to, like spam the chat box in order to like get something to send, like, you know, several messages at once, then when the condition comes back, discord pushes all of those out at once, and then the moderation what slaps them for it, because they posted like, way too many messages way too quickly. It's something that they can't do anything about.

Moreover, as P7 noted, it's not always the user's connection but also sometimes Discord servers or the bot itself lagging:

We've had false positives because of the bots lagging actually. So it received all the messages at the same time and muted like 20 people at the same time. Yet they were talking normally.

5.3 Having Multiple Bots for Moderation

Many participants also noted how they typically have multiple bots in server for moderation purposes because some bots may be better than others at certain moderation actions or because of the high demand for moderation, if their main moderation bot goes down, i.e., the provider hosting the bot crashes, then they need another moderation bot quickly or else they have to temporarily close the server.

In regards to having multiple bots for niche purposes in addition to a main bot, P1 notes:

[The bot we use] Crosslink is mainly in addition to [the main bot we use] Zeppelin because Zeppelin has the overall moderation and Crosslink just provides the additional sensor for all possible dangerous links.

P6 reported similar:

Zeppelin does like 99% of the heavy lifting. Beemo gets used like once every few months, when there's a big raid and Crosslink I imagine we'll get a bit more used because we get actually, like I said, that community, we have a challenge with people posting lots of links, and possibly posting lots of files.

In regards to issues with bots downtime, P8 noted:

MEE6 and Dyno are both public bots as well and so one or the other can go down. So one or the other is a backup for the other, you know what I mean. If Dyno goes down, MEE6 can take over and if MEE6 goes down, you know, the latter.

P3 also notes how their server plans to shift away from public bots, which due to demand, have more downtimes, because of a constant need for bots' moderation capabilities:

By moving away from like public bots and things like that… we can basically guarantee 100% uptime, which is really important for us, obviously, because when bots go down, we are fairly limited in what we can do. In the past, we have locked down a server a couple times because both [bots we rely on] went down. And we want to prevent that as much as possible.

The need for constant uptimes of bots reflects the "always fast" nature of real-time environments. People are always talking and always interacting on the server from all around the world so moderation teams are always in constant need of help when moderating.

7. Conclusion

Most automatic content moderation systems in general use filter-based algorithms to curate content; in other words, the algorithms used to remove or flag content compare content against some database containing keywords, regular expressions, etc. Every participant noted how their bot uses some sort of keyword filtering in order to moderate content. While automatic content moderation in real-time relies heavily on frequency-based algorithms, filter-based algorithms are just as important. The same cannot be said for non-real-time environments: non-real-time environment focus primarily on filter-based approaches to moderation algorithms while real-time has to handle new challenges.

Overall, real-time communities like those in Discord could not exist at scale without bots. As P1 noted: “[The bot,] it’s the heart of your server. If the heart breaks, it’s basically a rampage.” Ten out of the twelve participants noted explicitly how they could not do their jobs without bots to help. Furthermore, as P3 noted, real-time environments make moderators’ jobs time-sensitive:

We take action within seconds. And we have to think about what is appropriate within seconds. Otherwise, lots of people already see it, or isn't relevant anymore. If someone is spamming the channel, you want to take action immediately and not a minute later. So yeah, that's such a different way of moderating a platform…

These time sensitive needs introduces new challenges to moderators, such as raids, and new challenges to bot developers in addressing the issues which arise in real-time environments, such as frequency-based algorithms.

References

321: Auto Moderation in Discord. (n.d.). Discord. Retrieved March 17, 2021, from https://discord.com/moderation/1500000178701-321:-Auto-Moderation-in-Discord

451: Reddit-x-Discord. (n.d.). Discord. Retrieved March 17, 2021, from https://discord.com/moderation/1500000179901-451:-Reddit-x-Discord

About Discord. (n.d.). Discord. Retrieved March 17, 2021, from https://discord.com/company https://discord.com/safety/360043712132-How-we-investigate

Braun, V., & Clarke, V. (2012). Thematic analysis.

Cai, J., & Wohn, D. Y. (2019). Categorizing Live Streaming Moderation Tools: An Analysis of Twitch. International Journal of Interactive Communication Systems and Technologies (IJICST), 9(2), 36-50.

How we Investigate. (n.d.). Discord. Retrieved March 17, 2021, from

Jhaver, S., Birman, I., Gilbert, E., & Bruckman, A. (2019). Human-Machine Collaboration for Content Regulation: The Case of Reddit Automoderator. *ACM Trans. Comput.-Hum. Interact.* 26, 5, Article 31 (July 2019). https://doi.org/10.1145/3338243

Jiang, J. A., Kiene, C., Middler, S., Brubaker, J. R., & Fiesler, C. (2019). Moderation Challenges in Voice-based Online Communities on Discord. *Proc. ACM Hum.-Comput. Interact*. 3, CSCW, Article 55 (November 2019). https://doi.org/10.1145/3359157

Kiene, C., & Hill, B. M. (2020, April). Who Uses Bots? A Statistical Analysis of Bot Usage in Moderation Teams. In Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems (pp. 1-8).

Kiene, C., Jiang, J. A., & Hill, B. M. (2019). Technological Frames and User Innovation: Exploring Technological Change in Community Moderation Teams. *Proc. ACM Hum.-Comput. Interact*. 3, CSCW, Article 44 (November 2019), https://doi.org/10.1145/3359146

Lunden, I. (2020, December 17). *Discord confirms raising $100M at a valuation of $7B*. TechCrunch. https://techcrunch.com/2020/12/17/filing-discord-is-raising-up-to-140m-at-a-valuation-of-up-to-7b/

Zhang, A. X., Hugh, G., & Bernstein, M. S. (2020, October). PolicyKit: Building Governance in Online Communities. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (pp. 365-378).