Toxicity in the Decentralized Web and the Potential for Model Sharing[†]

HARIS BIN ZIA, Queen Mary University of London, United Kingdom
ARAVINDH RAMAN, Telefonica Research, Spain
IGNACIO CASTRO, Queen Mary University of London, United Kingdom
ISHAKU HASSAN ANAOBI, Queen Mary University of London, United Kingdom
EMILIANO DE CRISTOFARO, University College London, United Kingdom
NISHANTH SASTRY, University of Surrey, United Kingdom
GARETH TYSON[*], Hong Kong University of Science & Technology, Hong Kong

The "Decentralised Web" (DW) is an evolving concept, which encompasses technologies aimed at providing greater transparency and openness on the web. The DW relies on independent servers (aka instances) that mesh together in a peer-to-peer fashion to deliver a range of services (*e.g.* micro-blogs, image sharing, video streaming). However, toxic content moderation in this decentralised context is challenging. This is because there is no central entity that can define toxicity, nor a large central pool of data that can be used to build universal classifiers. It is therefore unsurprising that there have been several high-profile cases of the DW being misused to coordinate and disseminate harmful material. Using a dataset of 9.9M posts from 117K users on Pleroma (a popular DW microblogging service), we quantify the presence of toxic content. We find that toxic content is prevalent and spreads rapidly between instances. We show that automating per-instance content moderation is challenging due to the lack of sufficient training data available and the effort required in labelling. We therefore propose and evaluate *ModPair*, a model sharing system that effectively detects toxic content, gaining an average per-instance macro-F1 score 0.89.

1 INTRODUCTION

Social platforms such as Facebook and Twitter are amongst the most popular websites in the world. Despite their huge success, criticism for their role in disseminating toxic content (*e.g.* hate speech) is mounting. This is a thorny issue that could pose a threat to freedom of speech [3]: the monolithic nature and lack of competition of these platforms gives them full discretion over the activities that are allowed to take place. This has raised a number of regulatory concerns and triggered widespread political debate [68].

In reaction to this, a growing movement referred to as the "Decentralised Web" (DW) has emerged. The goal of the DW is to decentralize power and control away from the major centralised tech giants. As such, these projects have striven to create a more open environment that champions freedom of speech, whilst simultaneously disincentivising toxicity by giving users greater control over their own online communities. The DW consists of a range of platforms that offer decentralised equivalents to mainstream services. Some of the most popular DW applications include Pleroma and Mastodon [44] [57] (Twitter-like micro-blogging services), Diaspora [25] (a Facebook-like social network), and PeerTube [56] (a YouTube-like video hosting software). These DW platforms

†Published in the Proceedings of the 2022 ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS'22). Please cite accordingly.
[*]Also with Queen Mary University of London.

Authors' addresses: Haris Bin Zia, Queen Mary University of London, United Kingdom, h.b.zia@qmul.ac.uk; Aravindh Raman, Telefonica Research, Spain, aravindh.raman@telefonica.com; Ignacio Castro, Queen Mary University of London, United Kingdom, i.castro@qmul.ac.uk; Ishaku Hassan Anaobi, Queen Mary University of London, United Kingdom, i.h.anaobi@qmul.ac.uk; Emiliano De Cristofaro, University College London, United Kingdom, e.decristofaro@ucl.ac.uk; Nishanth Sastry, University of Surrey, United Kingdom, n.sastry@surrey.ac.uk; Gareth Tyson, Hong Kong University of Science & Technology, Hong Kong, gtyson@ust.hk.

have several unique properties: (*i*) They are made up of independently operated and moderated communities located on different servers (called *instances*), which any new administrator can setup; (*ii*) They enable users, who must sign-up to specific instances, to own their data – in fact, some users choose to fork their own instance to keep complete control of their data; and (*iii*) They allow users to interact locally (within instances) as well as globally (across instances) via the so-called "Fediverse" — this involves instances interconnecting in a peer-to-peer fashion, allowing their users to communicate (referred to as *federation*). Through these novel features, these independent servers collaborate to offer a globally integrated platform atop of an entirely decentralised infrastructure.

Despite its novelty, this model creates interesting challenges for *toxic content moderation* (§2). Namely, whereas centralised services (like Twitter) have the resources to actively moderate content by hiring moderators and refining automated classifiers using large data pools, DW administrators have limited resources and only control the data in their own instance. Furthermore, whereas many major services rely on third-party commercial and centralised moderation APIs [4] (that tag their posts), this goes against the vision and philosophy of the DW. Even if this were possible, an administrator can only moderate their own instance — it is possible for content from poorly moderated instances to "spread" to other instances through the peer-to-peer federation links. Of course, this all assumes that administrators even have similar definitions of "toxic". Thus, addressing the challenge of toxic content moderation has proven to be labour-intensive and impractical for many instance operators. A recent example of this occurred when the `gab.social` instance [83] joined the Fediverse, rapidly spreading hate speech to other instances [2]. We argue that addressing this problem is vital for ensuring the success of DW applications.

With this in mind, we identify four critical questions: (*i*) How much toxicity exists in the DW? (*ii*) How does toxic material spread across DW instances via federation? (*iii*) Is it possible for DW instances to train their own automated classification models to reduce the manual load on administrators? And (*iv*) How can administrators cooperate to improve content moderation and reduce their workload?

To address these questions, we present the first large-scale study of toxicity in the DW and propose *ModPair*, a collaborative approach to better enable instances to automate content moderation in a decentralised fashion. We do so from the point of view of one of the largest decentralised microblogging networks, *Pleroma*. We first present a measurement study of Pleroma (§4), gathering data from 30 unique Pleroma instances (§3), covering more than three years and 9.9M toots[1] from 117K users. We confirm that extensive toxic content *is* present in the DW. We identify 12.15% of all toots as toxic. Furthermore, we show that toxic content *does* spread across Pleroma: 26 out of the 30 instances receive an average of at least 105K remote toxic toots through federation. In fact, Pleroma instances receive more toxic content from remote instances than they generate locally. This makes it impractical for individual administrators to manually flag toxic toots at such a scale.

Driven by these observations, we explore the potential of decentralised automated content moderation in the DW (§5). We start by assessing the ability of administrators to train local models to automatically tag local toots as toxic *vs.* non-toxic. We confirm that it *is* possible to build such local models, attaining an average macro-F1 score of 0.84 across all instances. However, we find that such models struggle to accurately tag remote content due to divergent linguistic features. For example, whereas a model trained using `my.dirtyhobby.xyz` data gains a macro-F1 score of 0.95, it attains an average of just 0.69 when applied to toots from other instances. This makes it difficult to rely on such models to detect toxicity in incoming toots from remote instances, without requiring administrators to tag thousands of more toots.

[1]A toot is equivalent to a tweet in Twitter.

To improve moderation, we therefore propose *ModPair* (§6) — a system for collaborative moderation where instances share partially trained models to assist each other. This has the benefit of implicitly sharing annotations across instances, without compromising the privacy of administrators by asking them to expose their own moderation tags. To enhance knowledge transfer, *ModPair* leverages semantic similarity of instances by pairing instances with similar topical characteristics. Specifically, each instance uses its own small set of (annotated) toxic toots to build local models. Instances then predict which remote models are most likely to perform well on their own local content, creating an ensemble to improve their own classification performance. We show that ModPair can correctly identify the top three best performing remote models 83% of the time, gaining an average per-instance macro-F1 score 0.89. Although here we focus on text-based toxic content moderation, our methodology can equally apply to other types of decentralised content classification, *e.g.* image classification (§8).

2 BACKGROUND & MOTIVATION

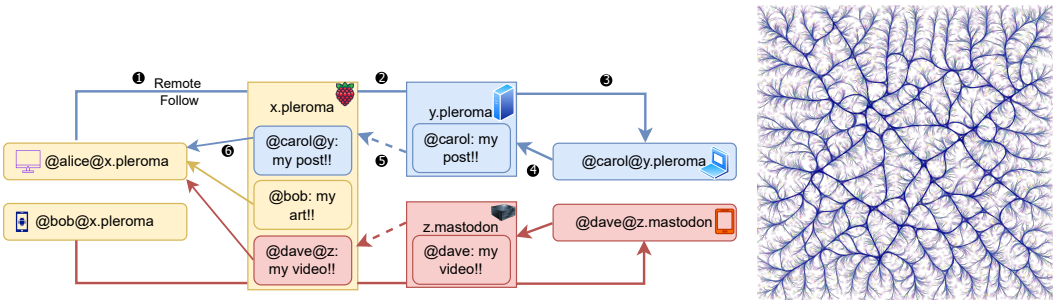
What is the Fediverse? The Fediverse is a network of independently hosted and interconnected "Decentralised Web" servers (*i.e.* instances). In this model, no single entity operates the entire infrastructure. Instead, instances collaborate (aka "federate") in a peer-to-peer fashion to collectively offer various types of services (*e.g.* microblogging, file sharing, video streaming). This federation is performed using the W3C ActivityPub [1] protocol, which allows instances to subscribe to objects provided by each other. The nature of these objects varies based on the specific application in question. For example, whereas Pleroma (a microblogging platform) exchanges messages, PeerTube (a video sharing platform) exchanges videos. This allows these server instances to form a decentralised network of content exchange. There are around 30 open source platforms that are built with ActivityPub. Servers installing any of these software packages are interoperable and can communicate with each other irrespective of the service they provide.

What is Pleroma? Pleroma is a DW open source *microblogging* service that runs in the Fediverse and the largest decentralised social platform (next to Mastodon [59]). Unlike centralised online social networks, anyone can run a Pleroma instance which will then operate independently under the control of its administrator. The Pleroma software is a lightweight web server written in Elixir, using the Phoenix framework with a database backed with PostgreSQL. Accordingly, administrators can deploy Pleroma on a wide range of hardware, ranging from Raspberry Pis to high-end compute servers. Both administrators and regular users can then access these instances using any web browser (via HTTP).

Each Pleroma instance has a unique domain name and users must sign up to a specific instance to gain access to the wider Pleroma network. Often, each instance will be themed around a given topic, *e.g.* users interested in arts might join `imaginair.es`, whereas programming might join `pythondevs.social`. Once a user has created an account on an instance, they can follow other accounts from the same instance or, alternatively, users on other instances that also host Pleroma.[2]

In Pleroma, "toots" are the equivalents of "tweets" in Twitter and much of the functionality (*e.g.* the ability to follow users and 'like" toots) works in a similar fashion. That said, there are clear differences between Pleroma and prior microblogging services: Pleroma (*i*) provides no ranking and recommendation algorithm instead toots are displayed chronologically; (*ii*) has no algorithm to recommend followees, instead new connections rely on searching an already known user through

[2]In fact, they can follow users from any other Fediverse instances that use the ActivityPub protocol (*e.g.* Mastodon, Peertube, *etc.*).



(a) (b)

Fig. 1. (*a*) Example federation workflow. (*b*) A graph representation of how instances exchange toots in the Fediverse. Nodes are instances, and links indicate that two instances are federated.

the search functions or exploring the instances to find like-minded users; (*iii*) is a community-oriented platform: each instance supports specific interests or topics and users can register on the instance that is better matched to their own tastes.

What is Federation in Pleroma? Once a user joins an instance, they can follow (*i*) other users on the same instance; or (*ii*) remote users from other instances. The latter action creates a federation link between two instances (*i.e.* federation). Thus, remote toots are retrieved by the local server (using ActivityPub) and presented to the local users on their timelines.

Figure 1a illustrates an example of how these toots are subscribed to and retrieved across the federated network. Consider three instances `x.pleroma`, `y.pleroma`, `z.mastodon` with users [Alice, Bob], Carol, and Dave, respectively. Let us assume `*.pleroma` hosts Pleroma and `*.mastodon` hosts a Mastodon microblogging service. ActivityPub allows Alice to follow Carol, who is a user on a different instance. While this is not possible with traditional platforms (*e.g.* a user on Twitter cannot follow a user on Facebook), the Fediverse allows this flexibility. This is done by Alice performing a remote follow request to Carol which involves the following steps. ❶ Alice makes a request to her local instance (`x.pleroma`) to follow Carol (on instance `y.pleroma`). ❷ The request is forwarded to `y.pleroma`. ❸ The remote instance (`y.pleroma`) informs Carol that Alice is now following her. Thereafter, whenever ❹ Carol posts a toot on her instance (`y.pleroma`), ❺ it gets pushed to Alice's instance (`x.pleroma`). Finally, ❻ when Alice logs in, the toot will appear on her timeline. Note, Alice can also view the video posted in `z.pleroma` as Bob (also from `x.pleroma`) remotely follows Dave (from `z.mastodon`). This federated approach results in a complex network, where instances are linked as a result of the following relationships of their user base.

Challenges in Toxic Content Moderation. Administration in the Fediverse is decentralised: each instance decides which toots are considered toxic *vs.* non-toxic. Thus, prior centralized approaches to moderation, where a single administrative entity (*e.g.* Twitter) has full control, no longer applies. This greatly complicates moderation as toots generated on one instance can easily spread to another instance, even if those two instances have wildly different viewpoints. For example, imagine `z.mastodon` allows the posting of pornographic content, yet `x.pleroma` does not. An explicit post by Dave –acceptable in his instance (`z.mastodon`)– may easily spread to `x.pleroma` where it is not acceptable. Given the huge scale of the Fediverse and the voluntary nature of most administrators, manual moderation of large quantities of content is not feasible. This would makes it difficult for `x.pleroma` to moderate all incoming posts from `z.mastodon`. Complicating this is the fact that instance administrators rarely wish to upload content to centralised moderation APIs,

as this naturally undermines the nature of the DW (and incurs extra costs). To give a preliminary sense of the scale of this federation network, Figure 1b presents the network representation of the federation links formed between the instances in our dataset (later explained in §3). Preventing the spread of toxic content with this dense and complex instance interconnectivity is challenging due to the ability for content to spread across the federated links.

3 DATASET AND METHODOLOGY

We start by describing our data collection and toxicity labeling methodology. Although the Fediverse contains a large number of diverse services, we focus on Pleroma (microblogging platform). We do this because it is one of the largest by both content and user counts. Our data collection follows three key steps: (*i*) Discovering a set of Pleroma instances to measure, including its federation network; (*ii*) Gathering the full set of toots from each instance; and (*iii*) Labeling each toot as toxic or non-toxic. For the latter, we use *toxicity* annotations from Jigsaw Perspective [4], due to its widespread uptake and well-understood definition.

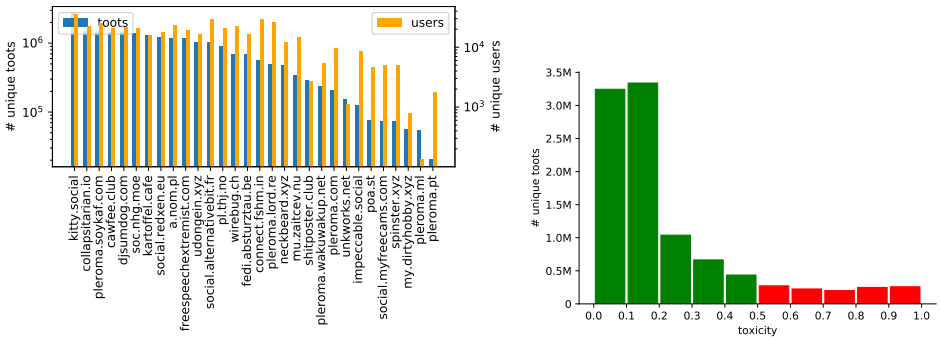
Discovering Instances. We first need to identify the domains of Pleroma instances deployed around the globe. To do this, we crawled a list of Pleroma instances from `the-federation.info/pleroma` on December 15, 2020. This yielded 729 unique instance domains. We then expanded this list by recursively capturing the instances that these instances federate (*i.e.* its list of remote instances it has previously connected to). This is done using each instance's Peers API[3] between December 2020 and January 2021. This API endpoint returns a complete list of instances that any instance has federated with during its lifespan. In total, we identify 1360 instances.

Collecting User Data & Toots. Next, we collect all public toots from the identified instances. For this, we gathered all toots using their Public Timeline API.[4] This API endpoint returns all public toots on the instance. Each toot includes information about the author, text content, associated media, timestamp, number of likes, number of reblogs, and any self-tagged content warnings. The latter is voluntarily added by the toot author to notify future viewers that the toot may contain (subjectively judged) sensitive material. In total, 5.4% of toots in our dataset are self-tagged with warnings.

To gather this data, we build and execute a multi-threaded crawler to gather all prior toots made before January 22, 2021, from all responsive instances. This covered 713 (out of 1360) instances. The primary reason for failures among the remaining instances was that many instances were not reachable (31.5%), and some (12.3%) had zero toots. The remaining instances did not make their toots publicly available and we made no attempt to circumvent the publicly available data restrictions. Finally, we gather associated user profile information, namely the full list of each account's followers and followees. Note, due to the distributed nature of the instances, we parallelised our crawler across several servers and implement a set of precautions, like rate limiting on API requests, to not burden the instances.

Toxicity Labels. This paper focuses on exploring the spread of toxicity on Pleroma. Hence, we label toots in our dataset using Google Jigsaw's Perspective API [4]. Our choice of Perspective API is motivated by similar and recent measurement studies of the other social platforms like 4chan [54] and Voat [53]. We apply the same classification model and toxicity definition for comparability. Perspective defines toxicity as "a rude, disrespectful, or unreasonable comment that is likely to make people leave a discussion". Internally, Perspective trains BERT-based models [24] on millions of comments from online forums such as Wikipedia and The New York Times where each comment

[3] <instance.uri>/api/v1/instance/peers
[4] <instance.uri>/api/v1/timelines/public



(a) (b)

Fig. 2. (a) Distribution of toots and users on Pleroma instances (note log scale on Y-axis). (b) Number of toots in different toxicity intervals.

is tagged by 3-10 crowdsourced annotators for toxicity. For a given toot, Perspective returns a score (between 0 and 1) for its toxicity. This offers an estimate of the fraction of human moderators who would label the content as toxic. Following [53, 54, 60, 66], we consider a *toot* to be toxic if its toxicity score is greater than 0.5 (and vice versa). This represents a moderately toxic score, but we opt for this to capture a broad range of problematic content and behavior. We label a *user* as toxic if the average toxicity of their toots is greater than 0.5 (and vice versa). For completeness, we also repeat our later experiments with a stricter toxicity threshold (0.8) to find that it provides similar results, as summarised in the Appendix A.

Unfortunately, due to rate limitations, it is not possible to label *all* toots in our dataset. Hence, we select the 30 largest instances based on the number of toots they contain (see Figure 2a). Overall, these 30 instances contain 9,927,712 unique toots (of which 1,394,512 were local toots) posted by 116,856 unique users (of which 8,367 were local users) between 1 January 2017 and 22 January 2021. This represents 55% of the total toot count for that period. The federated graph showing the subscriptions of these 30 instances is shown in Figure 1b. These 30 instances are connected to many further instances (on average 1,511). This implies that the annotated toots in our dataset include many toots that have been imported from other instances.

In §5 we use these Perspective annotations in lieu of human labels provided by administrators of each instance. A limitation of this methodology is that each instance will obviously have annotations using a consistent and standardised definition (as dictated by Perspective). This is not necessarily representative of where individual instance administrators may moderate on differing criteria. To address this, we later introduce randomised noise into the annotations on a per-instance basis, to reflect discrepancies between administrators' annotation styles.

4 CHARACTERISING TOXICITY IN PLEROMA

Considering recent media reports of toxic activity in the DW [67], we start by presenting a general overview of the toxicity in Pleroma. We use this to quantify and motivate the importance of moderation.

4.1 Overview of Toxicity

The toxicity labels obtained from Perspective range between 0 and 1. Figure 2b presents the distribution of toxicity scores across all toots. In line with studies of other social platforms [53, 54],

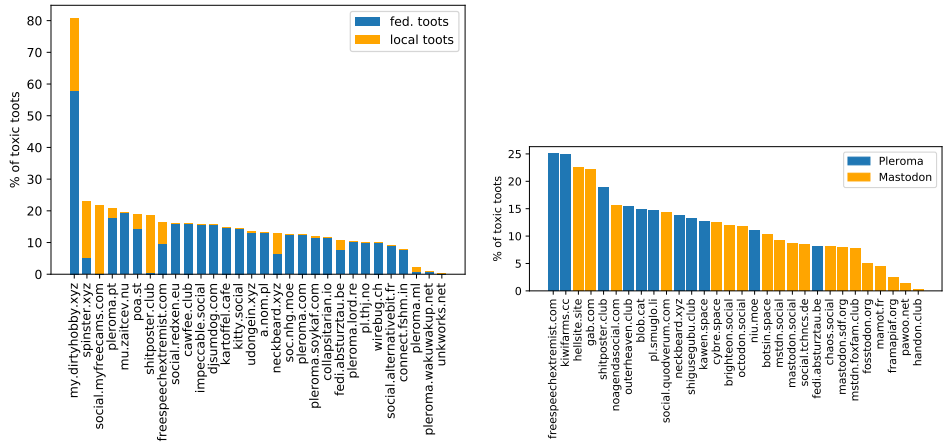


Fig. 3. Percentage of toxic toots (a) on Pleroma instances split by local and federated toots (b) from top-30 federated instances with the largest number of federated toots split by the origin of the federated toots.

the majority of toots in our dataset are non-toxic. On average, we find the amount of toxic content is significantly less than that observed on other online fringe communities such as 4chan (37%) [54] and Voat (39.9%) [53]. That said, we still find a large portion of toxic toots (12.15%).

We conjecture that this may be driven by certain instances spreading large volumes of toxic content. Thus, Figure 3a presents the percentage of toots on each instance that score above 0.5 toxicity. We find that 80.7% of toots on `my.dirtyhobby.xyz` are classified as toxic. This adult-themed instance is classified as toxic primarily due to the abundance of sex-related conversations. Note, this also highlights the diversity of toxic content types considered in centralised moderation APIs. In contrast, we observe that the majority of other instances have between 7–23% of toxic toots, confirming a wide range of toxicity levels. We do discover a small number of exceptions though, with 3 instances generating under 3% toxic toots: `pleroma.ml` (2.2%), `pleroma.wakuwakup.net` (0.8%), `unkworks.net` (0.3%). This confirms that the DW *is* composed of a variety of instance types, with certain instances generating a significant share of toxicity.

4.2 Spread of Toxicity

A unique aspect of the DW is *federation*, whereby content is imported from one instance to another. This poses a particular challenge for local administrators, who have little control over the remote instances where content is created.

Local* vs. *Federated Toots. To test if this concern is legitimate, we compare the toxicity of local *vs.* federated toots (*i.e.* toots that have been retrieved from a remote instance). Figure 3a plots the percentage of local *vs.* federated toxic toots on all 30 instances. In line with our intuition, we see that federated toots constitute the most significant chunk of toxic content on 26/30 of the instances. In contrast, only four instances have more toxic content generated locally (rather than imported via federation). Interestingly, this suggests that, on average, users tend to follow and import content that is more toxic than the locally generated one.

Next, recall that our dataset includes federated content that has been retrieved from instances outside of Pleroma (the W3C ActivityPub [1] protocol allows Pleroma instances to interoperate with

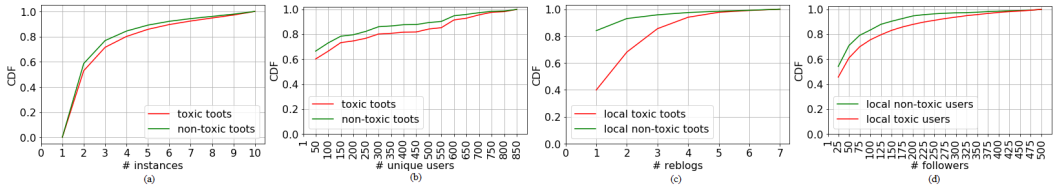


Fig. 4. CDF of (a) the number of instances federated toots reach split by toot toxicity, (b) number of users federated toots reach split by toot toxicity, (c) number of reblogs of toxic and non-toxic toots, (d) number of followers of toxic and non-toxic users.

various other DW-enabled platforms such as Mastodon). In total, our 30 instances have federated with 5,516 other instances from the wider Fediverse. This allows us to inspect the composition of toxic *vs.* non-toxic toots contributed by these other platforms. To this end, we extract the set of instances (from the 5,516) with at least 10% of their toots classified as toxic. This results in 1,489 non-Pleroma instances (27%) being extracted. Figure 3b plots the percentage of toxic toots from these federated instances with the largest number of federated toots. We observe a mix of both Pleroma and Mastodon (the two most popular DW microblogging services). For instance, we see prominent controversial Pleroma instances like `kiwifarms.cc`, an instance that manages various forms of group trolling, harassment, and stalking. We also observe several large Mastodon instances like `gab.com`, famed for hosting hate speech material [2, 83]. Interestingly, not all federated instances are equally harmful in absolute terms. We observe a significant number of instances that contribute a relatively small number of federated toots (< 100 per instance), even though a significant fraction of them are toxic. This confirms that federation *is* a significant challenge for per-instance moderation: regardless of how well an administrator moderates their own instance, it is possible for millions of remote (toxic) toots to be retrieved by users from other instances that may have very different policies. The openness of DW federation exacerbates this further, by allowing different platforms (*e.g.* Pleroma, Mastodon, PeerTube) to interoperate. Thus, any solutions cannot rely on remote instances adhering to identical practices, as they may be running distinct software stacks.

Reach of Toxic Toots. After showing how toxic content flows across instances, we now study how many instances and users this content spreads to. Figure 4a shows the number of instances that toots are replicated onto (via federation). The distributions are extremely similar, with a Kolmogorov-Smirnov p-value of 0.85. However, we do see that toxic content has a marginally higher probability of reaching a larger number of instances (than non-toxic content). In total, 52% of toxic toots are federated on at least one other instance, and, on average, toxic toots are replicated onto three instances *vs.* two for non-toxic toots.

This, however, may be misleading as different instances have different user populations sizes. For example, a toot being replicated onto 10 instances, each with one user, has less reach than a toot being replicated onto a single instance with a million users. To examine the audience reach in terms of the number of users the content hits, Figure 4b plots the number of unique users each toot reaches. We calculate this based on the number of registered users on each instance. Here, we observe clearer trends, with a Kolmogorov-Smirnov p-value of 0.24. This confirms that toxic toots reach noticeably larger audiences. 87% of non-toxic toots reach more than 1 user, compared to 92.3% for toxic toots. On average, toxic toots appear on the timelines of 1.4x more users.

4.3 Characterising Local Toots & Users

We next seek to explore the characteristics of local toots and users that are toxic *vs.* non-toxic. For this analysis, we therefore exclude all federated toots. In total, this leaves 1,394,512 unique local

Topic	Representative words	Distribution in all toots (%)	Distribution in toxic toots (%)	Toxic toots per topic (%)
General conversation	time, like, know, date, message	29.5	28.5	18.8
Profinity	fuck, wtf, ass, shit, holy	8.8	22.0	46.2
Dark web	dark, net, deep, box, web	7.4	4.4	11.2
Sex talks and online cams	online, love, video, cum, guys	7.0	7.9	21.7
Computers, e-games and tech.	game, linux, work, windows, play	7.0	4.9	13.3
Greetings and compliments	hope, good, day, morning, nice	6.9	4.4	12.1
Fediverse	post, server, mastodon, pleroma, instance	6.7	4.0	11.4
NSFW content	jpg, nsfw, source, tits, porn	5.4	0.9	3.3
COVID and economy	coronavirus, million, economics, pandemic, markets	4.9	1.5	6.0
Loli content	png, image, loli, screenshot, husky	4.7	1.6	6.5
Politics	trump, election, vote, biden, america	4.4	7.2	31.4
Humam rights (esp. gender)	women, men, gender, white, rights	3.7	9.1	46.3
Free speech and societal issues	governance, people, speech, crowd, free	3.6	3.6	18.7

Table 1. Topics in local toots as determined by CombinedTM and interpreted using pyLDAvis with their overall distribution, distribution in toxic toots, and percentage toxic toots per topic.

toots (of which 17.6% are toxic). These are posted by 8,367 unique local users (of which 12.7% are considered toxic). By focusing solely on local toots, we can better understand the characteristics of each individual instance's user base.

Reblogs. We first inspect the *reblog* rate of toxic toots (a reblog is equivalent to a retweet on Twitter). We conjecture that toxic toots are likely to get more reblogs than non-toxic ones. To explore this, we analyze the number of reblogs that toxic *vs.* non-toxic toots receive. Figure 4c presents the CDF of the number of reblogs observed. Indeed, we see that toxic toots gain substantially more reblogs. 60% of toxic toots get more than one reblog, compared to only 16% of non-toxic toots. This trend indicates that interest and uptake in toxic material are consistently greater. On average, toxic toots get 140% more reblogs than their non-toxic counterparts. This confirms the virality of toxic content and helps explain the greater spread of toxic toots.

Followers. Pleroma allows its users to follow other users across the Fediverse. In addition to toots, we also have information about each user's follower list. We next inspect whether there is a relationship between the number of followers a user has and the toxicity the account produces. Figure 4d plots the CDF of the number of followers of toxic and non-toxic users. Recall, we define a user as toxic their average toxicity score exceeds 0.5.

We observe that toxic users tend to have more followers, with a Kolmogorov-Smirnov p-value of 0.96. On average, toxic users have 142 followers compared to just 70 for non-toxic ones. One possible explanation is that this may be driven by differences in the frequency of toots (as the users who toot frequently are more likely to be viewed). To test this, we compute the toot:user ratio for toxic *vs.* non-toxic users. This actually shows a contrary trend. On average, non-toxic users have 188 toots per user compared to just 28 for toxic users. This suggests that our population of non-toxic users are actually more active. Therefore, we conjecture that these patterns may be driven by the higher reblog rates for toxic toots (see Figure 4 (c)), thereby increasing the exposure of such accounts.

Distribution of Topics. Until now, we have analysed toots regardless of their instance. However, users on different instances may hold discussions on very diverse topics with varying levels of toxicity. This may play a large role in defining and identifying toxicity. Thus, we analyze the topics found in local toots using topic modeling to get a more granular view [58].

For this, we use CombinedTM from Contextualized Topic Models (CTM) [11], which combines contextual embeddings with the bag of words to make more coherent topics. The number of

topics was chosen using a grid search over model coherence [65] and the model with the highest coherence was selected. In addition, we use pyLDAvis [71] to interpret topics and identify topic overlaps and similarity. Our final topic model results in 13 topics. We assign each toot with the most probable topic as predicted by the topic model. Table 1 lists the final topics, their overall distribution, distribution in toxic toots, and percentage of toxic toots per topic.

We observe that instances participate in diverse discourse ranging from sex to politics. Interestingly, the make-up of these conversations (in terms of toxicity) differs across each topic. Unsurprisingly, general everyday conversations form the largest portion of both overall (29.5%) and toxic (28.5%) toots. We also see several contrasting topics, where they contribute a larger share of toxic toots as compared to the overall distribution. For instance, human rights contribute just 3.7% of toots yet constitutes one of the most significant portions of toxic content (9.1%). In our dataset, this topic primarily covers gender issues, with strongly worded dialogue throughout. Similar comments can be made for Profanity (8.8% overall *vs.* 22.0% of toxic toots) and Politics (4.4% overall *vs.* 7.2% toxic). The former is not surprising as, by definition, profane toots are classified mainly as toxic. However, it is perhaps more worrying to see the significant density of toxic behaviour when discussing politics. As our dataset covers the 2019 United States Presidential Elections, we find extensive discussion about people such as Trump and Biden. This highly polarising topic triggered substantial confrontational and abusive language in our dataset. We again emphasise that our annotations are based on Perspective, providing a unified definition of toxic across the instances. In practice, we highlight that individual instance administrators may have differing views on what they personally consider toxic.

Topics* vs. *Instances. As topics may not be evenly distributed across all instances, we also analyse the distribution of topics on individual instances. We conjecture that some of the above trends may be driven by a subset of controversial instances. For example, a larger number of toots on a given topic could simply be generated by a single highly active instance. Figure 5a presents the percentage of toots belonging to each topic on a per-instance basis. We see that the topics discussed across instances *do* vary. Other than the general day-to-day discussions that occur almost everywhere, most instances prefer some topics more than others. For example, Not Safe for Work (NSFW) content is popular on `my.dirtyhobby.xyz` and `neckbeard.xyz`, whereas human rights (particularly gender issues) is discussed heavily on `spinster.xyz`. This flags up noticeable challenges for automating the detection of toxic content on the instances. This is because many toxic content models fail to generalize well beyond their target environment [75, 76]. Thus, applying models trained on human rights discussions to gender issues may not transfer well.

To further probe into the popularity of these topics, we also analyze the reblogs of toots belonging to each topic. Figure 5b shows the CDFs of the number of reblogs of toots split by topic. We observe that Loli content (sexualised anime material) receives more reblogs than any other topic, followed by NSFW toots. Again, this further confirms the virality of sensitive content on Pleroma instances.

5 EXPLORING AUTOMATED MODERATION

The previous section has confirmed the presence of large volumes of toxic material and its propensity to spread further and faster than non-toxic content. This is a challenge for administrators who must moderate not only their own instance's toots but also any federated material imported by their userbase. As the use of centralised moderation API (*e.g.* Perspective) undermines the decentralisation goals of the DW (as well as introducing cost, privacy and overhead issues), we argue that operating locally trained classification models is the only way forward. Thus, we next explore the potential of deploying automated content moderation on instances.

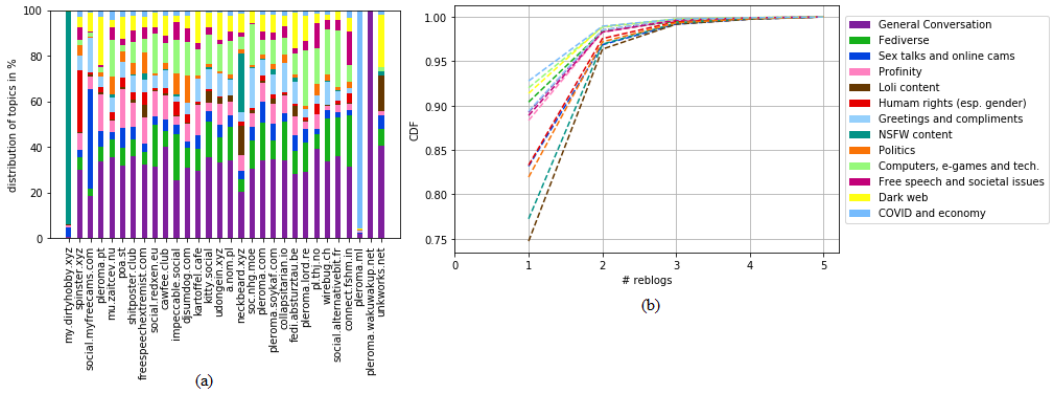


Fig. 5. (a) Distribution of topics on each instance, and (b) CDFs of the number of reblogs of toots belonging to each topic.

5.1 Experimental Methodology

As instances do not have in-built content classification tools, we propose and evaluate several potential architectures. We first take the toxicity annotations listed in §3 to train local models for each instance. These annotations include both the Perspective labels (recall that 12.15% of all toots are toxic) *and* the self-tagged content warnings for each toot (recall that 5.4% of all toots contain these). Note, we use the Perspective labels in lieu of human decisions made by an instance's administrator.

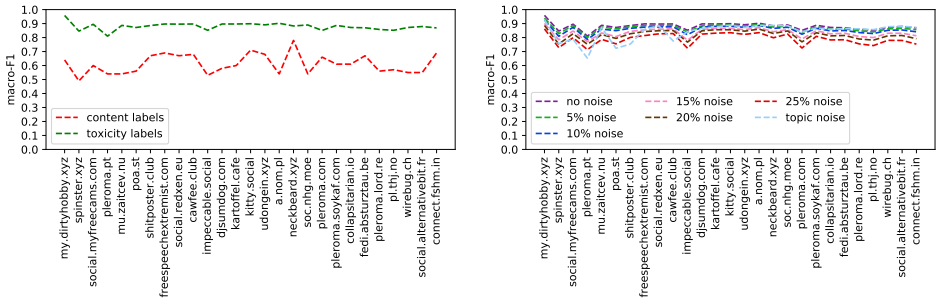
We next train models for each instance using their respective toot datasets. Recall that the data of each instance includes both the toots from the local users as well as the federated toots from remote users followed by the local ones (see §2). Numerous text classifiers could be used for this purpose. Due to the resource constraints of Pleroma instances (many run on Raspberry Pis), we choose the methodology used by two heavily cited seminal works [21, 82]. Namely, we rely on a Logistic Regression (LR) classifier with bag-of-words features [9], implemented using Scikit-learn [55]. We also experimented with an SVM classifier and obtained equivalent results (as measured using Student's t-test with $p > 0.05$). We report these additional experiments in the Appendix.Note, we acknowledge that this simple model although resource-light has limitations *e.g.* it discards word order and context and in turn could not differentiate between the same words differently arranged ("you are stupid" vs "are you stupid").

5.2 Performance Across Instances

First, we compare the performance of the models trained on the two different label schemes (content warnings *vs.* Perspective). We train per-instance models based on all the toots available to it, with an 80:20 split stratified by respective labels between training and testing data. Our goal is to estimate the feasibility of using content warnings to help automate the annotation process.

Content Warning Labels. First, we use the self-tagged content warnings as annotations. Figure 6a shows the macro-F1 scores of the classifiers for each instance. Overall, the results are not promising. The majority of the instances have a macro-F1 score less than or equal to 0.6, with only two instances exceeding 0.7.

We attribute this inefficiency to label noise introduced by inter-observer variability, *i.e.* different users perceive sensitivity differently, which results in uncertainty of labels. Also, we observe that these self-tagged content labels often do not necessarily correspond to typical interpretations of



(a) (b)

Fig. 6. Macro-F1 scores of classifiers trained on (a) content labels vs toxicity labels for each instance, (b) toxicity labels with varying percentages of noisy labels for each instance.

toxicity. For instance, we observe users adding content warnings to their toots that reference news articles pertaining to war. To evidence this better, we calculate the inter-agreement between the content warnings and the toxicity labels from Perspective: the Cohen's Kappa is just 0.01. In other words, there is only a small (random) chance that agreement exists between the two labels.

Perspective Toxicity Labels. The above indicates that using content warning labels is *not* suitable for training local models. Hence, we next repeat the previous training process for each instance using the Perspective labels instead. Figure 6a presents the macro-F1 scores of classifiers for each instance. The classifiers trained on Perspective labels show a significant improvement over those trained using content warnings. All the instances have a macro-F1 score greater than 0.8, with an average (calculated over all instances) of 0.84. This is 40% more than the average of classifiers trained on content warnings. This is reasonable as the Perspective labels exhibit far greater consistency than the self-tagged warnings. We therefore confirm that it *is* viable to train local models using data labeled with a consistent labeling scheme that can support and semi-automated content moderation for the administrators.

Variations in Toxicity Labels. One limitation of the above methodology is that the use of Perspective implies that annotations across all instances are identical. For example, if two instances have the same text in a toot, those two toots would be allocated identical Perspective labels. This does not necessarily reflect reality, as individual administrators may have different perceptions of what is "toxic". Administrators may also make mistakes or simply not exhibit consistent views across time.

To assess the effect of annotation inconsistency, we emulate mistakes that might take place in the annotation process. For each instance, we generate five new training sets with different noise levels (5% to 25%) by randomly flipping the labels of x% of the toots (where $x = [5, 25]$). We then repeat the above training process with these noisy labels.

Figure 6b presents the macro-F1 scores of the classifiers trained with varying degrees of noise in the labels. We observe an expected gradual decrease in performance as the noise in the labels increases. However, even with 25% of noisy labels, the average (calculated over all instances) macro-F1 score decreases only by 11.9%. This gives us confidence that it is feasible to build these local models, even in the presence of mistakes and inconsistency.

Whereas the above emulates mistakes, it does not reflect topical differences between annotation policies per-instance. Specifically, we expect that instances may have more systematic differences

based on the theme of their instances. For example, an instance dedicated to sharing adult content is unlikely to tag sexual-related material as toxic. To assess the impact of this, we generate a new topic-based training set for each instance. For each instance, we select the topic that is the most popular among its users (excluding "General Conversation", see Figure 5a). We then whitelist all toots in that topic, and label them as non-toxic. For example, we select NSFW content for `my.dirtyhobby.xyz` and Human rights for `spinster.xyz`.

Figure 6b presents the macro-F1 scores of the classifiers trained using this topic-based set. We see that these models perform far better than the prior experiments that introduced random noise. For these topic-based noisy models, the average performance decreases only by 4.7%. This is largely because the local training sets retain consistency on what is and what is not toxic.

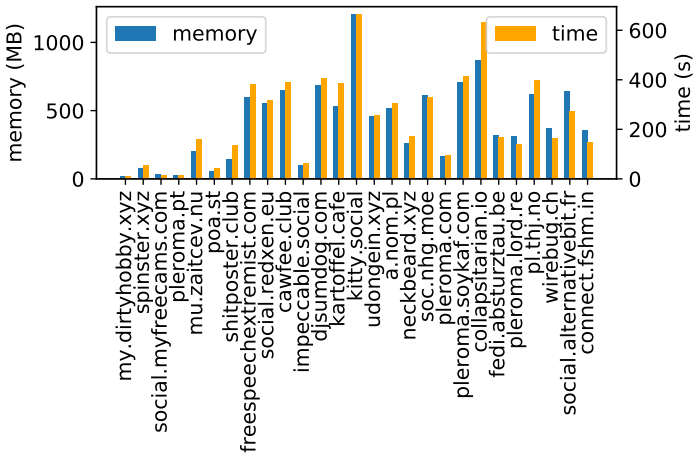


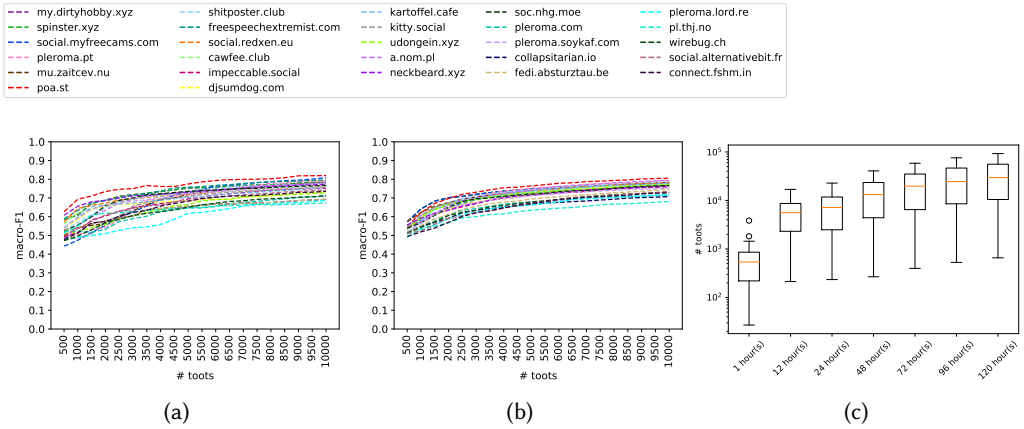
Fig. 7. Memory & Time consumption of training local models on Pleroma instances.

Computational Time and Cost. Due to the lightweight nature of each instance, for completeness, we finally plot the computational costs for training the models. We train each model on a machine with 128 GB of RAM with 16 cores (note, we did not use any GPUs). Figure 7 shows the memory footprint of training each model, as well as the time taken. We see a wide variety of values, largely incumbent on the number of training toots within the instance. For example, `kitty.social` takes 662.3 seconds (containing 196,6617 toots), whereas `my.dirtyhobby.xyz` takes just 8.0 seconds (containing 29,987 toots). On average, training takes 241 seconds per model. Importantly, we see that, on average, training only requires 409MB of RAM, suggesting this is well within the capacity of even lightweight hosted instances (*e.g.* the Raspberry Pi 4 has 8 GB RAM).

5.3 Annotation Feasibility

The prior experiments have a key assumption. The 80:20 split assumes that an administrator has time to annotate 80% of toots on their instance. In practice, this is probably infeasible due to the voluntary nature of most Pleroma administrators. Moreover, a newly created instance may only have annotated a tiny number of toots in its early days. Thus, we next evaluate how many toots each admin must label to attain reasonably good performance.

For this, we generate new training sets of different sizes. For each instance, we extract the first n toots (where $n = [500, 10000]$ in intervals of 500) in the timeline, and train 20 models (1x per set of size n). We also generate a second set, where we extract n random toots from across the entirety of



(a) (b) (c)

Fig. 8. Macro-F1 scores of classifiers (a) trained on first n toots, (b) trained on random n toots. (c) Distribution of toots accumulated on all instances at different time steps (note log scale on Y-axis).

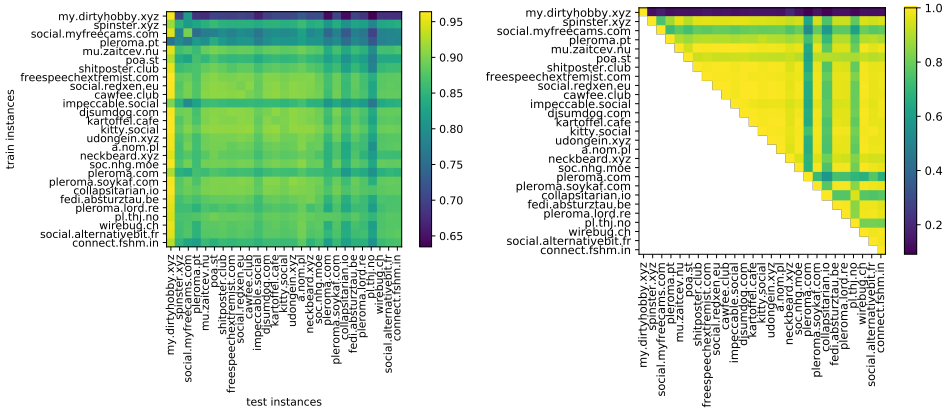
the timeline of each instance. These two sets represent the cases where (*i*) Administrators dedicate their time to annotating all toots for the initial period of operation; and (*ii*) Administrators allocate occasional time to annotate a small subset of toots across the entire duration of the instance's lifetime. In both cases, we repeat our prior training on each of these datasets and compute the macro-F1 on the test set of each instance.

Figure 8a shows the results generated from the first n toots, whereas Figure 8b shows the results for n toots randomly selected from across the entire timeline. In both cases, the X-axis depicts the size of the training set. We see similar results for all configurations. We observe an expected steady improvement in performance, with larger training sets (plateauing after around 6K posts). Although the trends are roughly similar for all instances, we do observe a range of performance and some outliers. For example, the models trained on `poa.st` achieve a relatively high macro-F1, starting with 0.62 on the first 500 toots and reaching a maximum of 0.81 on the first 10,000 toots. On average, instances attain a macro-F1 of 0.52 on the first 500 toots and 0.75 on the first 10,000. The averages remain roughly the same when the classifiers are trained on the same number of toots selected randomly.

Unfortunately, these results show that instances, on average, need more than 10K labeled posts to get a reasonably good classification performance (0.80 macro-F1). This means it will be difficult for instances to train and deploy their own local moderation models for two key reasons. First, different instances accumulate toots at different rates. To quantify this, Figure 8c presents a box plot showing the number of toots accumulated across different time periods. This is calculated by taking the average number of toots generated over these time periods on a per-day basis for each instance. We see a high degree of divergence (note the Y-axis log scale). For example, obtaining a training set of 10K toots would take `freespeechextremist.com` just 12 hours, in contrast to `social.myfreecams.com` which would take 16 days. Second, once this set of toots has been accumulated, significant manual effort is still required to label them. Finding ways to minimise these barriers is therefore vital.

6 MODPAIR: DECENTRALISED MODEL SHARING

We next explore collaborative approaches where instances can work together to improve automated toxic content moderation. Namely, we present *ModPair*, a system that facilitates the sharing of



(a) (b)

Fig. 9. (a) Comparison of models (in terms of macro-F1) trained on one instance (Y-axis) and tested on another instance (X-axis), (b) Comparison of cosine similarity across instances.

pre-trained models between instances. We will show that this can improve the accuracy of local models, with limited overheads for administrators.

6.1 The Potential of Model Sharing

As discussed above, a fundamental challenge is that training the classifier requires ground-truth labels (§5.3). While, for our experiments, we rely on the centralised Perspective API, this is unattractive and infeasible for decentralised instances (due to associated costs and the need to upload toots to centralised third parties). This leaves administrators to annotate toots manually, likely as part of their general moderation activities (*e.g.* upon receiving a complaint from a user about a given toot). This is both slow and laborious. We therefore argue that a potential solution is to allow better-resourced instances (in terms of annotations) to share their models with other instances. In this approach, models trained on one instance are "gifted" to another one.

To explore the potential of this, following the previous methodology (§5), we *train* a model on each instance using their toots. We then *test* that model on all other instances. This allows us to measure how transferable each model is across instances. Figure 9a presents the macro-F1 scores as a heatmap. The Y-axis lists the instances a model has been trained on, and the X-axis lists the instances a model has been tested on. We find that performance diverges across the instances. The least transferable model is that trained on `my.dirtyhobby.xyz`. Whereas it attains a macro-F1 score of 0.95 when tested on itself, it results in an average of just 0.69 across all other instances, *e.g.* just 0.63 when applied against `pl.thj.no`. That said, models trained on some other instances exhibit far greater transferability. The best performing model is that trained on `kitty.social`, which attains an average macro-F1 score of 0.89 across all other instances. Confirming our observations in §5.2, these trends are largely driven by the types of topics and material shared. For example, we find that 93.4% of `my.dirtyhobby.xyz` toots are identified as sex-related (see Figure 5a), whereas the remaining instances contain just 1.4% of such toots on average. This makes it hard to transfer such models, as their training sets differ wildly from those instances it is applied to. This confirms that decentralised model sharing *can* work effectively, but only in particular cases.

6.2 ModPair Design

The previous section shows that model sharing *can* work, yet performance is highly variable across instance-pairs due to the differing (linguistic) discourse. Thus, a clear challenge is identifying which instance models should be shared with other instances. This is not trivial, as it is not possible for instances to scalably inspect all content on each other.

ModPair Primer. We propose *ModPair*, a system to manage model exchange between instances in the Fediverse. ModPair runs on each instance, and executes the following three tasks: (*i*) *Step 1:* It automatically identifies instances suitable for model exchange based on their content; (*ii*) *Step 2:* It manages model exchange between the instances; and (*iii*) *Step 3:* It employs an ensemble to merge results from the top k models, predicted to be most transferable. ModPair continues to monitor activities across the Fediverse to constantly seek out better matching models. Although we implement ModPair in Pleroma, it is suitable for any other Fediverse platform that follows the same principles (*e.g.* PeerTube, Mastodon).

ModPair Design. Step 1: ModPair is first responsible for identifying instances with whom models should be exchanged. To achieve this, ModPair exploits *content similarity* between instances to predict pairs that should exchange their pre-trained moderation models. To achieve this, each instance locally generates a vector with each component corresponding to the *tf-idf* of words from the instance's toots. Specifically, we define this as:

$$tfidf(t,d) = tf(t,d).idf(t)$$

where

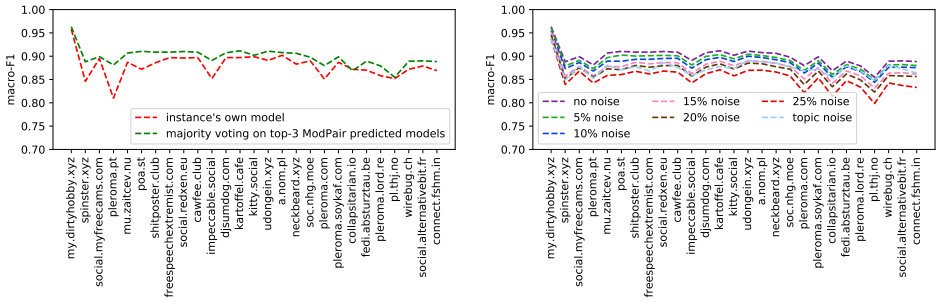
$$idf(t) = log[(1+n)/(1+df(t))] + 1$$

and n is the total number of toots in the instance and $df(t)$ is the number of toots in the instance that contain the term t. Instances then exchange these vectors, such that each instance can compute the *cosine similarity* between its own vector and all other vectors. This provides a measure of linguistic closeness between any two instances. To motivate this design choice, Figure 9b presents the empirical cosine similarity across the instances as a heatmap. By comparing this against Figure 9a, we see that content similarity *does* correlate with model performance, confirming the ability for distance to serve as a predictor for model performance.

Step 2: Once similarity has been locally computed, each instance must decide which other instances to download models from. To do this, ModPair uses a rank threshold, k, which stipulates the number of models an instance should retrieve. Specifically, an instance retrieves models from the k other instances that have the smallest cosine similarity (with its own local tf-idf vector). Note, downloading such models is very lightweight — using our dataset, we find that the largest model is just 7.7 MB, and the average size is only 3.2 MB per instance. The reciprocity of this process also helps incentivise participation. Step 3: Upon receipt of the pre-trained model(s), the instances can then use them to perform ensemble-based majority voting to classify future incoming toots.

ModPair Implementation. We have implemented ModPair as part of the Pleroma server software. In our implementation, each Pleroma instance exposes two new API endpoints to enable the above functionality: (*i*) one to publish their tf-idf vector (`<instance.uri>/api/v1/tfidf`); and (*ii*) one to share their local model (`<instance.uri>/api/v1/model`). Any instance requiring moderation models can therefore retrieve the vectors of other instances by communicating with their `tfidf` endpoint. By default, each instance will retrieve updated vectors from all other known instances each week.

ModPair Scalability. By default, ModPair selects the moderation models for an instance by computing the tf-idf cosine similarity with *all* other instances. Although straightforward, this may



(a) (b)

Fig. 10. Comparison of macro-F1 scores of (a) instance's own model *vs.* the majority voting on top-3 ModPair predicted models for each instance, (b) majority voting on top-3 ModPair predicted models with varying noise percentage for each instance.

result in scalability issues as the number of instances grows (due to the $O(n^2)$ complexity). For example, the 1360 Pleroma instances in our dataset (§3) would generate 1.8M transactions alone. Although ModPair rate limits its queries to avoid overwhelming remote instances, this workload is still undesirable.

To overcome the above challenges, ModPair introduces the optional concept of *pre-sampling* to select a subset of instances to exchange vectors directly with. This pre-sampling strategy is driven by the observation that often the best performing models are exchanged among instances that have a high degree of federation. Thus, on each instance, ModPair ranks all other instances by the number of followers in the target instance. This estimates the amount of social engagement between users on the two instances. We then pick the top f instances (default 5) with the most shared followers. Using this, each instance only retrieves the *tf-idf* vectors from these f instances and, therefore, only computes ModPair similarity amongst them. As before, we then select the top-3 most similar instances from this set of f options. Although this potentially misses higher-performing models, this massively reduces the number of retrievals and, consequently, the data exchanged between instances from 1.8M to 6,800 (assuming $f = 5$).

6.3 Evaluation Results

Performance of Model Selection. To evaluate the correctness of ModPair's selection of models, we calculate its Precision@k (or P@k) [19], where k is the similarity rank threshold ($k = 1, 3$). The precision reflects the proportion of the top-k ModPair predictions that are correct. In other words, this tests if ModPair selects the best models available. As an optimal baseline, we compare this against an oracle that pairs each instance with the three other instances whose models achieve the best performance on its local toots (as calculated in Figure 9a). Overall, ModPair achieves average (calculated over all instances) P@1 and P@3 scores of 0.74 and 0.83, respectively. In other words, 74% of the top-1 and 83% of the top-3 model selections of ModPair are correct. This confirms the viability of model sharing, where instances can request models from other similar instance(s) and use them for the moderation of their content. Importantly, this confirms the appropriateness of using lightweight tf-idf vectors to represent instance content.

Performance of Model Inference. We next test if ModPair improves the outcomes of the model inference. To evaluate the impact of ModPair on model performance, for each instance, we apply the top-3 ModPair predicted models (in a majority voting fashion) to its toots. This allows us to

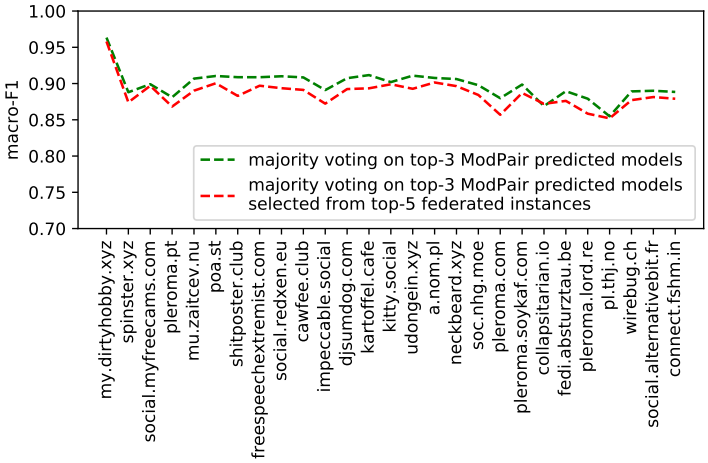


Fig. 11. Comparison of macro-F1 scores of the majority voting on top-3 ModPair predicted models selected from the pool of all instances *vs.* selected from the pool of 5 federated instances.

calculate the newly attainable macro-F1. We compare this with each instance's own model (*i.e.* locally trained). Figure 10a presents the results. We see that consistently ModPair outperforms local training across nearly all instances. On average, ModPair increases the macro-F1 by 5.9% compared to when each instance uses its own model alone (calculated over all instances). The results are also statistically significant as measured using Student's t-test with p = 0.01. Furthermore, ModPair also reduces the number of models (and consequently the number of annotated toots) required to moderate all the instances. For our collection of instances, it reduces the number of models to 10.

Performance of Model Inference with Noise. Recall, we conjecture that some training sets (and models) may contain noise, *e.g.* due to annotation mistakes by administrators (see §5.2). We next analyse the inference performance of ModPair in scenarios where individual instance models might be noisy (*e.g.* due to annotation inconsistency). To do so, for each instance, we apply the noisy versions (see §5.2) of the top-3 ModPair predicted models (in a majority voting fashion) to its toots and calculate the newly attainable macro-F1. Figure 10b presents the results. We observe that even with 25% noise induced into individual models, this results in limited performance degradation of just an average of 4.4% macro-F1 reduction (calculated over all instances). This is significantly lower than the degradation observed earlier when only using a single locally trained model (which obtained -11.9%). For the noisy models where we flipped the labels of toxic toots belonging to a particular topic, we observe the average degradation in performance of only 2.2%. This occurs because instances are being paired with others that have similar linguistic features. Therefore, this has little impact on overall performance because the paired instances are flipping similar topics. This further confirms the applicability of ModPair.

ModPair Scalability. The prior results are based on $O(n^2)$ model comparisons. Therefore we next repeat our experiments, using our scalable pre-sampling strategy that selects a subset of f instances to exchange $td - idf$ vectors with. Figure 11 compares the performance against the prior approach of retrieving all vectors. Confirming our intuition, we only observe a slight degradation (1.1%) in the average performance compared to when models are predicted from the pool of all instances. This occurs because instances often federate with others who discuss similar topics, therefore increasing the probability that their models may transfer well. Despite the small degradation in

performance, we also emphasise that the average performance is still 4.7% better than when each instance uses its own model (§5.2).

6.4 Discussion

Model sharing via ModPair allows instances to exchange models from other similar instances. This raises several issues worthy of discussion.

Privacy. ModPair enables instances to share the trained models (specifically share weights and biases of the trained models). In terms of privacy, this is better than sharing the raw data itself, preserving the privacy of toot text, as well as the annotations given by other administrators. Further, sharing of tf-idf vectors is privacy-preserving as they consider the words independently, hence it is not possible to reconstruct the original data. That said, studies show [39, 51], through reverse-engineering techniques, some important words or phrases used during the training can be identified. Although not explored here, we also note that there are protocols for computing cosine similarity and model sharing with privacy guarantees, which could be integrated with ModPair [28, 48].

Security in Adversarial Contexts. ModPair must also ensure it is not exploited by malicious parties. Most obviously, malicious instances could purposefully provide invalid models or tf-idf vectors to undermine other administrators. To mitigate this, ModPair's pre-sampling leverages follower relationships to better identify trusted similar instances. Nevertheless, this is a topic ripe for further exploration. We envisage prior work on peer-to-peer reputational models will be extremely useful here [35].

Fairness & Bias. There are chances that a model trained on a large instance with a large number of toots could be paired with a relatively small instance for model transfers. If such models are used widely, this could allow a small set of instances to bias the overall moderation process. Furthermore, this may "drown out" models from more fringe instances that might be useful for identifying specific forms of toxic content. While this is not necessarily a problem, as ModPair only pairs instances with similar interests, it may be necessary to adjust voting weights to prevent bias. We also plan to explore alternate ways of computing similarity that consider other factors such as the number of toots on each instance.

7 RELATED WORK

Decentralised Social Network Measurements. Prior work has extensively studied social networks with respect to their structure [5, 18, 38, 41, 42, 52, 73] and evolution [30, 37, 77]. The overwhelming majority of studies have been conducted on *centralised* social networks such as Twitter [46, 62], with only a handful of papers focusing on *decentralised* social networks. In [20], Datta *et al.* explore various motivations for decentralised social networking. Schwittmann *et al.* [69] analyze the security and privacy of decentralised social networks. Bielenberg et al. [12] shed light on the evolution of one of the first decentralized social networks, Diaspora, discussing its growth in terms of the number of users, and the topology of Diaspora's interconnected servers. Raman *et al.* [59] identify key challenges in the decentralised web, mainly related to network factors [36]. Finally, [84] authors collect data from Mastodon and explore several features such as the relationship network, placement of instances, and content warnings.

To the best of our knowledge, we are the first to measure the spread of toxicity on a decentralised social network. The independent and interconnected nature of instances makes this particularly different to prior studies of centralised platforms. Closest related to our work is [85], which gathered a dataset of content warnings from Mastodon (another DW platform). However, as shown in §5.2, we find that these are unsuitable for training classification models.

Toxic Content Classification. There have been a variety of works that attempt to build classifiers for the automatic identification of toxic content [74–76]. These include fundamental work on creating formal toxic content definitions [70] and compiling vital datasets [7, 22, 31]. Of particular interest are prior attempts to build text classifiers to automatically flag toxic material.

Deep learning has been widely used to classify toxic posts [8, 15]. The effectiveness of these mechanisms has been extensively evaluated [16, 34, 45, 76], alongside the design of techniques to circumvent such tools [27]. Badjatiya *et al.* [8] were amongst the first to show the benefits of training models to identify hate speech (using 16K annotated tweets). Rizos *et al.* [64] evaluate the performance data augmentation techniques applied on conventional and recurrent neural networks trained on Yahoo [78] and Twitter [21]. Wulczyn *et al.* [82] similarly use 100K user labelled comments from Wikipedia to identify the nature of online personal attacks. Others have shown that more traditional machine learning models such as n-grams and Logistic Regression [79] SVMs [72], Naïve Bayes, decision trees and random forests [21] also produce good results. There has also been work identifying controversial posts by using non-text features, such as user profile metadata [29, 63, 81].

Related to our work are recent efforts in the area of transfer learning [13, 24]. In terms of toxic content identification, these exploit pre-trained language models such as BERT, and perform fine-tuning to specialise the model [50, 80]. This subsequently 'transfers' semantic understanding from the general language model to the task in-hand. Although similar in concept, we take a very different approach to transferring knowledge between instances. Specifically, ModPair relies on model exchange and ensemble voting. This is because transfer fine-tuning [26] and even inference [23] are substantially more costly than lightweight models such as Logistic Regression (considering the limited capacity of most Pleroma instances). In our future work, we intend to explore the potential of using lightweight pre-trained models such as LadaBERT [43].

Moderation in Social Networks. There have been a number of studies that have applied the above classification models to measure toxic activities within social networks, including Twitter [14, 62, 79], Reddit [6, 16, 49], 4Chan [10, 54], as well as various fringe forums [22]. There have also been studies that focus on specific domains of toxicity, including anti-Semitism [78], cyberbullying [17], white supremacy [22], misogyny [31], and Islamaphobia [40]. More recently, Anaobi *et al.* [32] highlighted the content moderation challenges for the instance administrators in decentralised web. We contribute to this wider space, by focusing on characterising the spread of toxicity in a novel platform: Pleroma. Beyond these prior works, we also show how such content spreads across independently operated instances (a unique feature of the Fediverse). As part of this, we identify and quantify hitherto unknown challenges for moderation that result from the Fediverse's decentralised architecture. Finally, we propose and evaluate a solution, ModPair, that allows instances to automatically exchange models to support scalable decentralised moderation.

8 CONCLUSION & FUTURE WORK

This paper has examined toxicity in Pleroma, a popular Decentralised Web (DW) social platform. We have characterised the spread of toxicity on the platform, confirming that the federation process allows toxic content to spread between instances. We have further explored the challenges of moderating this process by building per-instance models. We found that whilst this can be effective, it comes with a heavy burden on administrators who must annotate toots. To reduce this burden and enable collaboration amongst instances, we presented ModPair, a system for pairing instances that can share pre-trained models. We showed that by exchanging models with just a small set of other instances, administrators can effectively collaborate to improve each others' detection accuracy. Further, we have shown how ModPair can be scaled-up by using pre-sampling to dynamically

select remote instances that are likely to have a close linguistic similarity. Our work contributes to the wider debate on online toxic behaviour, as well as offers tools that can support the growth of platforms with expanding importance within the Fediverse.

As part of our future work, we plan to expand our analysis to other DW platforms and investigate other behavioural attributes, including the time-variance of tf-idf vectors (*i.e.* as instances generate toots, their tf-idf representation will change with time). We also intend to experiment with alternative features (*e.g.* semantically-rich sentence embeddings [61]) as well as classification approaches (*e.g.* allowing more resource-capable instances to train and share LSTM [33] or BERT-based models [24]). As part of this, we are curious to investigate the feasibility of using other types of Federated Learning [47, 48] too. Finally, we are keen to better understand the discrepancies between the annotations performed by different administrators (*e.g.* in terms of how they locally define toxicity). Through this, we hope to gain further evaluative insight into how ModPair can be useful to administrators.

ACKNOWLEDGMENTS

This work is supported by EPSRC grants SODESTREAM (EP/S033564/1), AP4L (EP/W032473/1), EPSRC REPHRAIN "Moderation in Decentralised Social Networks" (DSNmod), and EU Horizon grant agreements No 830927 (Concordia) and No 101016509 (Charity).

REFERENCES

[1] 2018. ActivityPub W3C Recommendation. https://www.w3.org/TR/activitypub/.
[2] 2019. Decentralized social media platform Mastodon deals with an influx of Gab users. https://www.tsf.foundation/blog/decentralized-social-media-platform-mastodon-deals-with-an-influx-of-gab.
[3] 2021. Why free speech on-the internet isn't free for all. https://www.bloomberg.com/news/articles/2021-06-19/why-free-speech-on-the-internet-isn-t-free-for-all-quicktake.
[4] 2022. Perspective API. https://www.perspectiveapi.com/.
[5] Yong-Yeol Ahn, Seungyeop Han, Haewoon Kwak, Sue Moon, and Hawoong Jeong. 2007. Analysis of topological characteristics of huge online social networking services. In *Proceedings of the 16th international conference on World Wide Web*. 835–844.
[6] Hind Almerekhi, Bernard J Jansen, and Haewoon Kwak. 2020. Investigating toxicity across multiple Reddit communities, users, and moderators. In *Companion proceedings of the web conference 2020*. 294–298.
[7] Kofi Arhin, Ioana Baldini, Dennis Wei, Karthikeyan Natesan Ramamurthy, and Moninder Singh. 2021. Ground-Truth, Whose Truth?–Examining the Challenges with Annotating Toxic Text Datasets. *arXiv preprint arXiv:2112.03529* (2021).
[8] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep Learning for Hate Speech Detection in Tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion* (Perth, Australia) *(WWW '17 Companion)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 759–760. https://doi.org/10.1145/3041021.3054223
[9] Adam Berger, Stephen A Della Pietra, and Vincent J Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational linguistics* 22, 1 (1996), 39–71.
[10] Michael Bernstein, Andrés Monroy-Hernández, Drew Harry, Paul André, Katrina Panovich, and Greg Vargas. 2011. 4chan and/b: An Analysis of Anonymity and Ephemerality in a Large Online Community. In *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 5.
[11] Federico Bianchi, Silvia Terragni, and Dirk Hovy. 2020. Pre-training is a hot topic: Contextualized document embeddings improve topic coherence. *arXiv preprint arXiv:2004.03974* (2020).
[12] Ames Bielenberg, Lara Helm, Anthony Gentilucci, Dan Stefanescu, and Honggang Zhang. 2012. The growth of diaspora-a decentralized online social network in the wild. In *2012 proceedings IEEE INFOCOM workshops*. IEEE, 13–18.
[13] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* (2020).
[14] Pete Burnap and Matthew L Williams. 2015. Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making. *Policy & internet* 7, 2 (2015), 223–242.
[15] Rui Cao, Roy Ka-Wei Lee, and Tuan-Anh Hoang. 2020. DeepHate: Hate Speech Detection via Multi-Faceted Text Representations. In *12th ACM Conference on Web Science* (Southampton, United Kingdom) *(WebSci '20)*. Association

for Computing Machinery, New York, NY, USA, 11–20. https://doi.org/10.1145/3394231.3397890

[16] Eshwar Chandrasekharan, Umashanthi Pavalanathan, Anirudh Srinivasan, Adam Glynn, Jacob Eisenstein, and Eric Gilbert. 2017. You can't stay here: The efficacy of reddit's 2015 ban examined through hate speech. *Proceedings of the ACM on Human-Computer Interaction* 1, CSCW (2017), 1–22.

[17] Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Emiliano De Cristofaro, Gianluca Stringhini, Athena Vakali, and Nicolas Kourtellis. 2019. Detecting cyberbullying and cyberaggression in social media. *ACM Transactions on the Web (TWEB)* 13, 3 (2019), 1–51.

[18] Xu Cheng, Cameron Dale, and Jiangchuan Liu. 2008. Statistics and social network of youtube videos. In *2008 16th Interntional Workshop on Quality of Service*. IEEE, 229–238.

[19] Nick Craswell. 2009. *Precision at n*. Springer US, Boston, MA, 2127–2128. https://doi.org/10.1007/978-0-387-39940-9_484

[20] Anwitaman Datta, Sonja Buchegger, Le-Hung Vu, Thorsten Strufe, and Krzysztof Rzadca. 2010. Decentralized online social networks. In *Handbook of social network technologies and applications*. Springer, 349–378.

[21] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. In *Eleventh International AAAI Conference on Web and Social Media*.

[22] Ona De Gibert, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. 2018. Hate speech dataset from a white supremacy forum. *arXiv preprint arXiv:1809.04444* (2018).

[23] Lysandre Debut. 2019. Benchmarking Transformers: PyTorch and TensorFlow. https://medium.com/huggingface/benchmarking-transformers-pytorch-and-tensorflow-e2917fb891c2.

[24] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[25] Diaspora. 2010. https://diasporafoundation.org.

[26] Ben Dickson. 2020. The GPT-3 economy. https://bdtechtalks.com/2020/09/21/gpt-3-economy-business-model/.

[27] Ysabel Gerrard. 2018. Beyond the hashtag: Circumventing content moderation on social media. *New Media & Society* 20, 12 (2018), 4492–4511.

[28] Zakaria Gheid and Yacine Challal. 2015. An efficient and privacy-preserving similarity evaluation for big data analytics. In *2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*. IEEE, 281–289.

[29] Vicenç Gómez, Andreas Kaltenbrunner, and Vicente López. 2008. Statistical Analysis of the Social Network and Discussion Threads in Slashdot. In *Proceedings of the 17th International Conference on World Wide Web* (Beijing, China) *(WWW '08)*. Association for Computing Machinery, New York, NY, USA, 645–654. https://doi.org/10.1145/1367497.1367585

[30] Roberto Gonzalez, Ruben Cuevas, Reza Motamedi, Reza Rejaie, and Angel Cuevas. 2013. Google+ or Google-? Dissecting the evolution of the new OSN in its first year. In *Proceedings of the 22nd international conference on World Wide Web*. 483–494.

[31] Ella Guest, Bertie Vidgen, Alexandros Mittos, Nishanth Sastry, Gareth Tyson, and Helen Margetts. 2021. An expert annotated dataset for the detection of online misogyny. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. 1336–1350.

[32] Anaobi Ishaku Hassan, Aravindh Raman, Ignacio Castro, Haris Bin Zia, Emiliano De Cristofaro, Nishanth Sastry, and Gareth Tyson. 2021. Exploring content moderation in the decentralised web: The pleroma case. In *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies*. 328–335.

[33] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[34] Shagun Jhaver, Sucheta Ghoshal, Amy Bruckman, and Eric Gilbert. 2018. Online Harassment and Content Moderation: The Case of Blocklists. 25, 2, Article 12 (mar 2018), 33 pages. https://doi.org/10.1145/3185593

[35] Sepandar D Kamvar, Mario T Schlosser, and Hector Garcia-Molina. 2003. The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the 12th international conference on World Wide Web*. 640–651.

[36] Sebastian Kaune, Konstantin Pussep, Christof Leng, Aleksandra Kovacevic, Gareth Tyson, and Ralf Steinmetz. 2009. Modelling the internet delay space based on geographical locations. In *2009 17th Euromicro International Conference on Parallel, Distributed and Network-based Processing*. IEEE, 301–310.

[37] Ravi Kumar, Jasmine Novak, and Andrew Tomkins. 2010. Structure and evolution of online social networks. In *Link mining: models, algorithms, and applications*. Springer, 337–357.

[38] Jure Leskovec and Eric Horvitz. 2008. Planetary-scale views on a large instant-messaging network. In *Proceedings of the 17th international conference on World Wide Web*. 915–924.

[39] Scott M. Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 4768–4777.

[40] Walid Magdy, Kareem Darwish, Norah Abokhodair, Afshin Rahimi, and Timothy Baldwin. 2016. # isisisnotislam or# deportallmuslims? Predicting unspoken views. In *Proceedings of the 8th ACM Conference on Web Science*. 95–106.

[41] Gabriel Magno, Giovanni Comarela, Diego Saez-Trumper, Meeyoung Cha, and Virgilio Almeida. 2012. New kid on the block: Exploring the google+ social graph. In *Proceedings of the 2012 Internet Measurement Conference*. 159–170.

[42] Lydia Manikonda, Yuheng Hu, and Subbarao Kambhampati. 2014. Analyzing user activities, demographics, social network structure and user-generated content on Instagram. *arXiv preprint arXiv:1410.8099* (2014).

[43] Yihuan Mao, Yujing Wang, Chufan Wu, Chen Zhang, Yang Wang, Yaming Yang, Quanlu Zhang, Yunhai Tong, and Jing Bai. 2020. Ladabert: Lightweight adaptation of bert through hybrid model compression. *arXiv preprint arXiv:2004.04124* (2020).

[44] Mastodon. 2016. `https://joinmastodon.org`.

[45] Antonis Matakos, Evimaria Terzi, and Panayiotis Tsaparas. 2017. Measuring and moderating opinion polarization in social networks. *Data Mining and Knowledge Discovery* 31, 5 (2017), 1480–1505.

[46] Binny Mathew, Ritam Dutt, Pawan Goyal, and Animesh Mukherjee. 2019. Spread of hate speech in online social media. In *Proceedings of the 10th ACM conference on web science*. 173–182.

[47] H Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. 2016. Federated learning of deep networks using model averaging. *arXiv preprint 1602.05629* (2016).

[48] Fan Mo, Hamed Haddadi, Kleomenis Katevas, Eduard Marin, Diego Perino, and Nicolas Kourtellis. 2021. PPFL: privacy-preserving federated learning with trusted execution environments. In *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services*. 94–108.

[49] Shruthi Mohan, Apala Guha, Michael Harris, Fred Popowich, Ashley Schuster, and Chris Priebe. 2017. The impact of toxic language on the health of reddit communities. In *Canadian Conference on Artificial Intelligence*. Springer, 51–56.

[50] Marzieh Mozafari, Reza Farahbakhsh, and Noel Crespi. 2019. A BERT-based transfer learning approach for hate speech detection in online social media. In *International Conference on Complex Networks and Their Applications*. Springer, 928–940.

[51] W James Murdoch, Peter J Liu, and Bin Yu. 2018. Beyond word importance: Contextual decomposition to extract interactions from lstms. In *Proceedings of the International Conference on Learning Representations*.

[52] Seth A Myers, Aneesh Sharma, Pankaj Gupta, and Jimmy Lin. 2014. Information network or social network? The structure of the Twitter follow graph. In *Proceedings of the 23rd International Conference on World Wide Web*. 493–498.

[53] Antonis Papasavva, Jeremy Blackburn, Gianluca Stringhini, Savvas Zannettou, and Emiliano De Cristofaro. 2021. "Is it a Qoincidence?": An Exploratory Study of QAnon on Voat. In *Proceedings of the Web Conference 2021*. 460–471.

[54] Antonis Papasavva, Savvas Zannettou, Emiliano De Cristofaro, Gianluca Stringhini, and Jeremy Blackburn. 2020. Raiders of the lost kek: 3.5 years of augmented 4chan posts from the politically incorrect board. In *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 14. 885–894.

[55] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* 12 (2011), 2825–2830.

[56] PeerTube. 2018. `https://joinpeertube.org`.

[57] Pleroma. 2016. `https://pleroma.social`.

[58] Daniel Ramage, Evan Rosen, Jason Chuang, Christopher D Manning, and Daniel A McFarland. 2009. Topic modeling for the social sciences. In *NIPS 2009 workshop on applications for topic models: text and beyond*, Vol. 5. 1–4.

[59] Aravindh Raman, Sagar Joglekar, Emiliano De Cristofaro, Nishanth Sastry, and Gareth Tyson. 2019. Challenges in the decentralised web: The mastodon case. In *Proceedings of the Internet Measurement Conference*. 217–229.

[60] Elizabeth Reichert, Helen Qiu, and Jasmine Bayrooti. 2020. Reading between the demographic lines: Resolving sources of bias in toxicity classifiers. *arXiv preprint arXiv:2006.16402* (2020).

[61] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

[62] Manoel Horta Ribeiro, Pedro H Calais, Yuri A Santos, Virgílio AF Almeida, and Wagner Meira Jr. 2018. Characterizing and detecting hateful users on twitter. In *Twelfth international AAAI conference on web and social media*.

[63] Sebastián A. Ríos, Roberto A. Silva, and Felipe Aguilera. 2012. A Dissimilarity Measure for Automate Moderation in Online Social Networks. In *Proceedings of the 4th International Workshop on Web Intelligence & Communities* (Lyon, France) *(WI&C '12)*. Association for Computing Machinery, New York, NY, USA, Article 3, 9 pages. `https://doi.org/10.1145/2189736.2189741`

[64] Georgios Rizos, Konstantin Hemker, and Björn Schuller. 2019. Augment to prevent: short-text data augmentation in deep learning for hate-speech classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 991–1000.

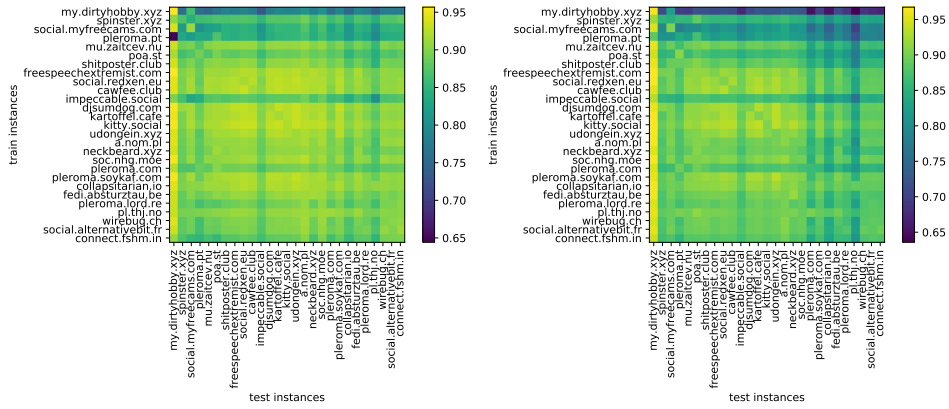
[65] Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*. 399–408.

[66] Paul Röttger, Bertram Vidgen, Dong Nguyen, Zeerak Waseem, Helen Margetts, and Janet Pierrehumbert. 2020. Hatecheck: Functional tests for hate speech detection models. *arXiv preprint arXiv:2012.15606* (2020).
[67] GREGORI SAAVEDRA. 2021. Terrorists are hiding where they can't be moderated. `https://www.wired.co.uk/article/terrorists-dweb`.
[68] Adam Satariano. 2021. Facebook Hearing Strengthens Calls for Regulation in Europe. `https://www.nytimes.com/2021/10/06/technology/facebook-european-union-regulation.html`.
[69] Lorenz Schwittmann, Christopher Boelmann, Matthäus Wander, and Torben Weis. 2013. SoNet–Privacy and replication in federated online social networks. In *2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops*. IEEE, 51–57.
[70] Amit Sheth, Valerie L Shalin, and Ugur Kursuncu. 2021. Defining and detecting toxicity on social media: context and knowledge are key. *Neurocomputing* (2021).
[71] Carson Sievert and Kenneth Shirley. 2014. LDAvis: A method for visualizing and interpreting topics. In *Proceedings of the workshop on interactive language learning, visualization, and interfaces*. 63–70.
[72] Sara Owsley Sood, Elizabeth F Churchill, and Judd Antin. 2012. Automatic identification of personal insults on social news sites. *Journal of the American Society for Information Science and Technology* 63, 2 (2012), 270–285.
[73] Amanda L Traud, Peter J Mucha, and Mason A Porter. 2012. Social structure of Facebook networks. *Physica A: Statistical Mechanics and its Applications* 391, 16 (2012), 4165–4180.
[74] Betty Van Aken, Julian Risch, Ralf Krestel, and Alexander Löser. 2018. Challenges for toxic comment classification: An in-depth error analysis. *arXiv preprint arXiv:1809.07572* (2018).
[75] Bertie Vidgen and Leon Derczynski. 2020. Directions in abusive language training data, a systematic review: Garbage in, garbage out. *PloS one* 15, 12 (2020), e0243300.
[76] Bertie Vidgen, Alex Harris, Dong Nguyen, Rebekah Tromble, Scott Hale, and Helen Margetts. 2019. Challenges and frontiers in abusive content detection. In *Proceedings of the third workshop on abusive language online*. 80–93.
[77] Bimal Viswanath, Alan Mislove, Meeyoung Cha, and Krishna P Gummadi. 2009. On the evolution of user interaction in facebook. In *Proceedings of the 2nd ACM workshop on Online social networks*. 37–42.
[78] William Warner and Julia Hirschberg. 2012. Detecting Hate Speech on the World Wide Web. In *Proceedings of the Second Workshop on Language in Social Media* (Montreal, Canada) *(LSM 2012)*. Association for Computational Linguistics, USA, 19–26.
[79] Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*. 88–93.
[80] Bencheng Wei, Jason Li, Ajay Gupta, Hafiza Umair, Atsu Vovor, and Natalie Durzynski. 2021. Offensive Language and Hate Speech Detection with Deep Learning and Transfer Learning. *arXiv preprint arXiv:2108.03305* (2021).
[81] Markus Weimer, Iryna Gurevych, and Max Mühlhäuser. 2007. Automatically assessing the post quality in online discussions on software. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*. 125–128.
[82] Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex Machina: Personal Attacks Seen at Scale. In *Proceedings of the 26th International Conference on World Wide Web*. 1391–1399.
[83] Savvas Zannettou, Barry Bradlyn, Emiliano De Cristofaro, Haewoon Kwak, Michael Sirivianos, Gianluca Stringini, and Jeremy Blackburn. 2018. What is gab: A bastion of free speech or an alt-right echo chamber. In *Companion Proceedings of the The Web Conference 2018*. 1007–1014.
[84] Matteo Zignani, Sabrina Gaito, and Gian Paolo Rossi. 2018. Follow the "Mastodon": Structure and evolution of a decentralized online social network. In *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 12.
[85] Matteo Zignani, Christian Quadri, Alessia Galdeman, Sabrina Gaito, and Gian Paolo Rossi. 2019. Mastodon content warnings: Inappropriate contents in a microblogging platform. In *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 13. 639–645.

A ADDITIONAL EXPERIMENTS

In addition to experimenting with a toxicity threshold of 0.5, we also conducted experiments with a much stricter threshold 0.8. In this case, we considered a toot toxic if its toxicity score is > 0.8 (and vice-versa). We performed N^2 experiments as shown in Figure 12a and the results were similar to that obtained in case of 0.5 threshold.

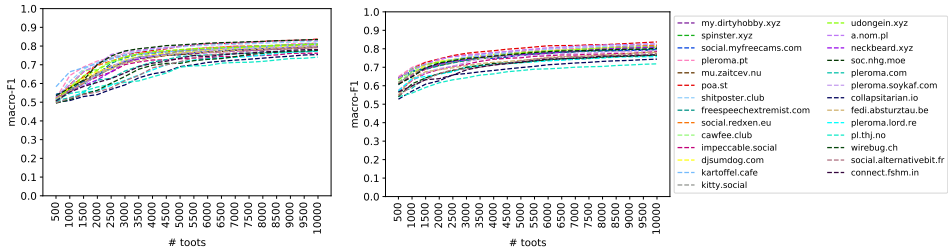
Furthermore, we also conducted experiments with a SVM classifier and, as shown in Figure 12b, results are comparable to Logistic Regression. These additional experiments further emphasize that our methodology and approach can be reused with other labeling schemes and classifiers.



(a) using toxicity threshold of 0.8.

(b) using SVM classifier.

Fig. 12. Comparison of models (in terms of macro-F1) trained on one instance (Y-axis) and tested on another instance (X-axis).



(a) With toxicity threshold of 0.8.

(b) With SVM classifier.

Fig. 13. Macro-F1 scores of classifiers trained on random n toots.