Username Squatting on Online Social Networks: A Study on X

Anastasios Lepipas
Department of Computing
Imperial College London, UK
a.lepipas20@imperial.ac.uk

Anastasia Borovykh
Department of Computing
Imperial College London, UK
a.borovykh@imperial.ac.uk

Soteris Demetriou
Department of Computing
Imperial College London, UK
s.demetriou@imperial.ac.uk

ABSTRACT

Adversaries have been targeting unique identifiers to launch typo-squatting, mobile app squatting and even voice squatting attacks. Anecdotal evidence suggest that online social networks (OSNs) are also plagued with accounts that use similar usernames. This can be confusing to users but can also be exploited by adversaries. However, to date no study characterizes this problem on OSNs. In this work, we define the *username squatting* problem and design the first multi-faceted measurement study to characterize it on *X*. We develop a username generation tool (UsernameCrazy) to help us analyze hundreds of thousands of username variants derived from celebrity accounts. Our study reveals that thousands of squatted usernames have been suspended by *X*, while tens of thousands that still exist on the network are likely bots. Out of these, a large number share similar profile pictures and profile names to the original account signalling impersonation attempts. We found that squatted accounts are being mentioned by mistake in *tweets* hundreds of thousands of times and are even being prioritized in searches by the network's search recommendation algorithm exacerbating the negative impact squatted accounts can have in OSNs. We use our insights and take the first step to address this issue by designing a framework (SQUAD) that combines UsernameCrazy with a new classifier to efficiently detect suspicious squatted accounts. Our evaluation of SQUAD's prototype implementation shows that it can achieve **94%** F1-score when trained on a small dataset.

CCS CONCEPTS

• **Information systems** → **Social networks**; • **Security and privacy**;

KEYWORDS

username squatting, social networks, impersonation, typo-mentions

ACM Reference Format:
Anastasios Lepipas, Anastasia Borovykh, and Soteris Demetriou. 2024. Username Squatting on Online Social Networks: A Study on X. In *ACM Asia Conference on Computer and Communications Security (ASIA CCS '24), July 1–5, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3634737.3637637

ASIA CCS '24, July 1–5, 2024, Singapore, Singapore

ACM ISBN 979-8-4007-0482-6/24/07.
https://doi.org/10.1145/3634737.3637637

1 INTRODUCTION

Users in online social networks (OSNs) are distinguished through unique identifiers or *usernames*. These identifiers are used for searching for and interacting with other users. Anecdotal evidence suggest that similar usernames might lead to confusion on the network. To make things worse, adversaries—who have been known to opportunistically hijack popular OSN accounts [65]—can deliberately weaponize this feature to impersonate popular or influential users to further their malicious campaigns and advertise content, products or services, spread propaganda, hate content or fake news, or harm influential individuals [80, 86].

Adversaries have targeted unique identifiers in the past using a range of squatting techniques with great success. For instance, traditional typo-squatting attacks target users who mistype a URL [108], mobile-app squatting attacks create mobile apps with names similar to popular apps to increase their installation counts and otherwise fool mobile users [43], and voice squatting attacks target phonetic similarities between the invocation names of voice assistant applications to hijack their communication with their users [56].

However, there is no systematic measurement of similar phenomena on OSNs. Prior works focused on analyzing and detecting fake accounts, content and bots on OSNs. These fail to study the extent of squatting on OSNs and mostly rely on expensive graph traversals to identify Sybil nodes [35] resulting in non practical detection methodologies. The rise of disinformation and fake accounts on OSNs already forced regulators to employ stricter policies—such as the *Code of Practice on Disinformation* [1] backed by the European Union's *Digital Services Act* [88]—to hold OSNs accountable for disinformation and fake accounts on their platforms. Therefore, it is timely and paramount to better understand squatting on OSNs and study techniques for tackling this problem.

Our work is the first which studies *username squatting* on OSNs. We define and characterize *username squatting*, and develop an efficient and effective methodology for identifying squatted accounts on OSNs. To achieve our first goal, we design a systematic measurement study for one of the most popular networks [69], *X* (previously known as Twitter [109]). Our study is structured around four measurement questions: (**MQ1**) Is *username squatting* a prevalent problem on *X*? If it is, then (**MQ2**) how does it contribute to online confusion; (**MQ3**) what are the username characteristics of the most problematic username variants, and (**MQ4**) what is the specific behavior of the squatted accounts (i.e. what is the *purpose* of the owner of such accounts)?

To aid our analysis, we developed a tool, UsernameCrazy, inspired by typo- and mobile app- squatting methodologies [108, 43] but tailored to the characteristics of OSN accounts, which uses 10 string generation strategies to efficiently construct more than 851,000 username variants for the top 97 (in 2019) popular *X* accounts [51], for which we collected more than 1.2 million tweets

in total. 83 out of 97 are still in top 100 most popular accounts in 2023 [8, 76]. Our measurement study led to a number of important findings. We found that tens of thousands of potentially confusing variants exist on the network; a large number of username variants of popular accounts are *mistakenly* mentioned (typo-mentioned) and that confusing variants appear high in the OSN's search recommendation list—and at times even *before* the verified account (see Section 4.2). We found that variants with a small edit distance are more likely to be mentioned and suggested. Both phenomena lead to increased attention for these confusing accounts which malicious actors can consequently exploit. As proof of this, we observed that a subset of these variants are already suspended which suggests that *username squatting* is indeed leveraged for malicious purposes. Moreover our three data collection snapshots across one year show that this is a growing problem as we found the number of suspended variants to increase over time. Analysing the squatted accounts, we show that thousands of squatted accounts which are currently *active* on the network are likely bots. A significant amount is involved in selling products, political goals or spreading fake news. Thus, as expected, the confusion generated through similar usernames and profiles tends to be exploited with malicious intentions.

Our study further shows that a large amount of squatted username accounts are involved in *previously unknown* impersonation attacks; To tackle this, *X* recently launched a premium service which allowed everyone paying a small fee ($8 per month) to be granted a blue check on their account [90]. Blue checks signal account verification and this feature was quickly and heavily manipulated by a swath of fake accounts to spread misinformation [47] and even damage companies' brands and stock prices [33, 21]. Companies reported to be withdrawing their ad campaigns from *X* due to the uncertainty of the platform's directions and to be pausing their *X* publishing plans for all their corporate accounts [39, 75, 25]. The feature was reverted and *X* claimed that blue checks will be granted only after manual verification [111]. The platform relaunched a modified version of their premium service which uses color–coded badges for different entities (organisations vs individuals etc.) along with an increased fee ($11 per month) if registered from a mobile device [90]. However, anecdotal evidence suggest that color badges are not a foolproof method, allowing adversaries to add cloned badges next to their profile name [41, 15] Also, manual verification is cumbersome and costly which is further exacerbated by the fact that OSNs currently operate with restricted resources [27, 29].

Our measurement study revealed that profiles sharing many similar features with the original account may try to exploit the created confusion and generate traction to their profile to promote products or services, or share potentially malicious links. This led us to our design question: (**DQ**) Can we design an effective and efficient framework for detecting suspicious username squatting attempts? Toward this, we leverage our previous observations to design **SQUAD**. SQUAD combines user name generation and online account discovery with the classification of suspicious squatted accounts. By ensuring a low false positive rate, we envision our framework to be an effective tool in scanning the OSN, saving time in manual screening and resulting in the more rapid discovery of suspicious profiles. A prototype of SQUAD trained on a small dataset reaches an F1-score of 94% in detecting suspicious accounts. We further applied SQUAD on popular and non-popular accounts and found that squatting targets popular users (55% of accounts squatting popular users are classified as malicious). Upon analyzing their activity we reveal two trends: (a) posting insecure URLs and at a lesser but non-negligible extend (b) trying to amplify their follower base. Our findings were responsibly disclosed (see Appendix H for details). SQUAD is available on GitHub [57]). Examples of impersonators, confusing variants, fake tweets and suspicious links, and additional results are available on our project's website [58].

Contributions. We summarize our main contributions below:

- *New techniques.* We propose UsernameCrazy, an efficient tool embodying new techniques for generating squatted versions of OSN usernames.
- *Findings.* We performed a measurement study and found that username squatting is a prevalent and growing problem, it can contribute to online confusion, and is used for malicious activity.
- *Detection Framework.* We design, implement, evaluate and apply SQUAD, a novel end-to-end framework that leverages username squatting as a strong signal to efficiently and effectively discover suspicious squatted accounts.

2 PROBLEM STATEMENT

What is username squatting? OSNs allow their users to create accounts with ease. Users can select any available string for their username[1], which can be used to uniquely identify them across the network. Other users can leverage those usernames to search, mention and interact with account owners. However, this poses a significant threat, as a malicious actor can easily select identifiers at account creation time which are confusingly similar to targeted entities or individuals for malicious purposes. Such techniques have been applied in the past to impersonate web domains (URL squatting [108, 85]), mobile apps (mobile app squatting [43]) and even voice personal assistant skills (skill squatting [56, 122]) with very high success rates. Similarly, a malicious actor could select a username which is similar with usernames of popular accounts in an attempt to impersonate the target account and/or spread confusion. We call this adversarial strategy *username squatting*. We will refer to squatted usernames as account usernames that are confusingly similar to popular account usernames (such as the ones of verified accounts). Similarly, we will refer to accounts that use squatted usernames as squatted accounts.

Definitions. We distinguish benign and suspicious accounts with the latter split into confusion and impersonation.

Benign. Benign accounts have an account name which is a username variant of an original account. However, these users either do not share a profile name, or their profile image is not similar to the original account, or they explicitly mention that they are a fan or parody users which by *X*'s policy is allowed [103].

Confusion. We define *confusion* accounts on OSNs to be squatted accounts which share a sufficient number of features with the original accounts and are suspicious (i.e., clearly not benign). Bot accounts are a subset of this class. Confusion can be exploited for malicious purposes (e.g., share links, fake news) and their impact

[1]There is no mention that username policies, in contrast with suspension policies [94, 93], are not the same globally.

can be amplified through features of OSNs (e.g., the search engine, likes, mentions). Figure 1 illustrates a scenario where a verified account on *X* (identified by the blue check on its name [100]) of a prominent news network mistakenly mentions a fake confusion account "cnnnbrk", which is *liked* and *retweeted* by numerous users.

Figure 1: Typo-mention: CNN mistakenly mentions *@cnnnbrk*, a squatted user of the official *@cnnbrk* profile.

Impersonation. We define accounts to be *impersonation* attempts if these have the same features as the original accounts *and* impersonate their *behavior* either through the tweets or information contained in the bio (e.g., fake verification blue check). Impersonations are a subset of the broader confusion class. While we focus on discovering the more general confusion accounts, we do find a number of impersonations as well.

3 MEASUREMENT STUDY DESIGN

3.1 Data Collection and Username Generation.

Ethics in Data Collection. We analyzed data available through *X*. We did not extract any personal opinions or other viewpoints linked to individuals as this could potentially be sensitive. In detail, a) all our collected raw data (profile data, tweets with mentions etc.) were captured using official APIs and b) all the unnecessary data that APIs return are discarded immediately. We also b) never share any information with 3rd parties and c) encrypt and keep in a local disk all the data after each experiment. Besides, we follow recommendations in [114] and mask the account names of individuals in the paper and do not include any tweets of individual accounts, including only public tweets from the top 97 most popular users.

Data Collection Methods. We focus our analysis on *X* due to its popularity. *X* has more than 353 million monthly active users in 2023 [69]. We select the 100 most popular accounts of *X* [51]. We call this dataset the '*Initial Seed*'. Out of those accounts, at the data collection time '@realdonaldtrump' had been suspended while '@aamir_khan' and '@arianagrande' had been deleted or deactivated. We exclude these accounts from our dataset and all measurements. This is because *X* API does not return information about suspended, deactivated and deleted accounts. This would pose a threat to validity in our methodology [98] as for the following search recommendation and amplification of confusion analysis the results would not include those related to these accounts. A challenge we had to overcome in crawling *X* stems from API request rate limits which the platform imposes to prevent denial of service (DoS). One could alternatively web crawl the platform using multiple registered accounts. Such an approach, has ethical concerns since it imposes a strain on the studied network while it is also less predictable and reproducible since the client side structure of the network might change at any point. In our study, we leverage *X*'s academic research product [2] track authorization [99] (2021). This allows us to freely access the full-archive search endpoint of *X* API (with access to public conversations and tweets posted back to 2006) with less strict rate limits [104].

Username Variant Generation. Our work focuses on studying usernames that might be confusingly similar with ones of already popular accounts. Prior works have leveraged seeded network graph traversals for identifying potentially fake or impersonation accounts [35, 116]. Such techniques are only practical when the target accounts are within a small distance from the account they are impersonating, which might not necessarily be the case with all username variants. In contrast, squatting techniques lend themselves well to our problem. Existing tools such as URLCrazy [108] and AppCrazy [43] even though not complete, they have been proven successful in generating complex string variations. Nonetheless their string generation models are tailored to their specific problems, that of squatting URLs and mobile app identifiers (package names), respectively. Hence, these tools suffer from several limitations because they are restricted to their domains (we refer to them as *prior models*). For example, they produce a large number of strings which are incompatible usernames due to the restrictions that OSNs set for valid usernames. Besides, neither of them supports useful generation patterns, like inserting digits at the beginning or end of a word which would be an easy way for an adversary to automate username variant creations. For clarity we defer a more concrete comparison with prior tools to Section 6.1.

To address the limitations of existing tools, we develop an end-to-end tool called **UsernameCrazy** to drive our experiments. UsernameCrazy takes as input a set of account usernames (the *Initial Seed*). For each account it employs 10 string generation models (we refer to them as *primitive models*) to produce valid username variants. For each variant, the generation model runs until the username reaches the maximum number of allowed characters [107]. The produced variants ('*Generated Seed*') are used in our measurement studies. UsernameCrazy generation models are categorized as follows a) Insertion models, b) Deletion models, and c) Misspellings. Figure 2 illustrates this taxonomy with examples. In Appendix A we briefly describe each model.

UsernameCrazy introduces several important enhancements to improve coverage of username variants compared to existing tools [108, 43]. Firstly, it employs four new generation models, important to username squatting, namely *number insertion*, *number deletion*, *underscore insertion* and *underscore deletion* (see the *blue-shaded* boxes in Figure 2). Secondly, each model is applied repeatedly (*model self-repetition*) on a username. For example, this allows UsernameCrazy's *Double Character Insertion* model to uniquely produce '@Jimmmyfalllon'. Thirdly, it supports *model stacking*: after applying a model on a username it takes all the generated variants and applies a second model on them. For example, the combination of *Vowel Insertion* and *Vowel Character Substitution* uniquely generates '@BearackObama', '@BoarackObama' etc. We note that i) during *model stacking* the models are *self-repeated* and ii) the generated usernames of the *primitive models* are not reproduced when *model self-repetition* is applied. We measured the contribution

[2] Academic Research Access was deprecated on March 2023 [91] but the full-archive search is still available through a paid *pro* or *enterprise* level account.

of each of the new features of UsernameCrazy to account discovery and compare it with what can be achieved with *prior models* [108, 43]. The results are presented in Table 1. In Section 6.1 we perform a more detailed analysis for each of UsernameCrazy's models.

Table 1: Account discovery comparison between prior string generation models, and UsernameCrazy's primitive models, primitive models with self-repetition, and primitive models with model stacking.

Accounts	Prior Models	Primitive Models	Self-Repetition	Model Stacking
Active	2,055	3,416	24,465	13,512
Suspended	436	676	6,656	3,029

3.2 Measurement Methods.

3.2.1 ***MQ1.*** *Measuring Prevalence.* To answer our first MQ, we apply UsernameCrazy on the *Initial Seed* with the most popular X accounts [51] ($n = 97$) to generate their username variants. We focus on popular accounts since these are more likely to be targeted by username squatting. Three independent raters went through the accounts and labelled them according to X's accepted account types [100]: a) *'Government'* (5 accounts), b) *'Companies, brands and organizations'* (5 accounts), c) *'News organizations and journalists'* (10 accounts), d) *'Entertainment'* (59 accounts), e) *'Sports and gaming'* (15 accounts) and f) *'Activists, organizers and other influential individuals'* (3 accounts). Conflicts were resolved with discussion. Then we search for each of the variants on X to find how many of them currently exist. We call such accounts *active*. To better understand if such username squatting can be exploited for malicious purposes, we use X's Academic API v2 (`User Lookup` [106]) to check if any of the generated variants were already suspended.

3.2.2 ***MQ2.*** *Measuring Squatted Account Effects.* To understand the impact of squatted accounts we focus on features of OSNs that allow users to interact with and find other users. These are likely to be affected by username squatting issues since they depend on users typing usernames of other users and any typos made will drive traffic to and amplify the impact of squatted accounts. In particular, we focus on the ability of users to *mention* and to *search* for other users. Both are popular features present in most OSNs.

User Mentions. We conduct experiments to measure whether username squatting affects the *mention* functionality. More interesting to our analysis are *typo-mentions*. We define a *typo-mention* as a mention of a username which was likely made by mistake. Such typo-mentions can drive more activity to the squatted account, amplifying their potentially malicious impact. To measure such typo-mentions we use the full-archive search endpoint (API v2 [104]) to search for the most recent 500 tweets that *mentioned* each username variant. These tweets can be *actual tweets*, *retweets* or *replies* and for each tweet, we collect the number of likes, retweets, quoted retweets and whether it can contain sensitive information such as links. Note that a *mention* of a username variant on its own does not necessarily indicate a typo-mention.

To further distinguish typo-mentions from purposeful mentions, we use the following observation: a user is more likely to have made a typo-mention, if the mentioner–mentionee are not related. Or conversely, if they are related, then a mention of a username variant was most likely on purpose. We can determine this relationship by looking at the distance between mentioner-mentionee in the social graph. In our experiments we set this distance to '1' due to practical limitations such as rate limits imposed by X API.

To evaluate how accurately the above method can capture typo-mentions we can randomly select a subset of the *actual tweets* and manually label them as *typo–mentions* and *purposeful–mentions* to create a ground truth set. Then we can apply our method which classifies a tweet as a *typo–mention* if the mentioner and mentionee are within one hop in the social graph or as a *not–typo–mention* otherwise to create a prediction set. An issue with random sampling is that we might end-up with an unbalanced dataset—the majority of the tweets could be cases of *purposeful–mentions*. To overcome this and collect a perfectly balanced dataset, we use purposive sampling [53, 71] instead: we randomly collect 100 *actual tweets* that are likely *typo-mentions (i.e where the mentioner and mentionee are* not *friends)* and another 100 *actual tweets* where the mentioner and mentionee are friends. This process resulted in 200 sampled tweets covering tweets from 80% of the users in our *Initial Seed*. Another issue we faced was that during labelling of the tweets we found that a number of the tweets are difficult to label. Therefore, we introduce a third ground truth label, '*difficult–to–declare*'. After labeling, the ground truth resulted in 51% *typo-mentions*, 38.5% *purposeful–mentions*, and 10.5% *difficult–to–declare*. We define a *true positive* as a tweet that is marked as *typo-mention* from both our method and the ground truth; false positive is a tweet that our method outputs as a *typo-mention* but in the ground truth is labelled otherwise; a true negative is a tweet that our method outputs as a *not–typo–mention* and in the ground truth that same tweet is labelled as either *purposeful–mention* or *difficult–to–declare*; and a false negative is a tweet that our algorithm outputs as a *not–typo–mention* and in the ground truth that same tweet is labelled as *typo–mention*. The results are summarized on Table 2. These correspond to 74/26/72/28/74/72.5 for TP/FP/TN/FN/Precision/Recall. Overall we observe that our method for detecting typo–mentions, even though simple, it can yield good results (> 70 on both precision and recall), which allows us to apply it to get an overall understanding of typo–mention prevalence on X.

Table 2: Evaluation of *Typo-Mention* measurement method.

		Actual Values		
		Typo-Mention	Purposeful-Mention	Difficult-to-Declare
Predicted Values	Typo-Mention	74	16	10
	Not-Typo-Mention	28	61	11

User Search Recommendations. Most OSNs make recommendations to users trying to either *mention*, *tag* or *search* for another user. We design experiments to gain evidence on whether such recommendations can exacerbate confusion due to username squatting. In particular, we use the `GET users/search` [102] method of X API, which takes as input a prefix of a username and returns the first 1000 matching results. We apply this request for each of the possible prefixes of a username (if a username has n characters, we issue n requests). We repeat this experiment for each user of the *Initial Seed* to examine whether the generated variants are ranked higher in the list of the recommended users than the respective original accounts and how often this phenomenon happens.

3.2.3 ***MQ3.*** *Username Variants Characteristics.* For MQ3, we aim to analyze the characteristics of squatted usernames to help us understand which ones are more prevalent or contribute to on the

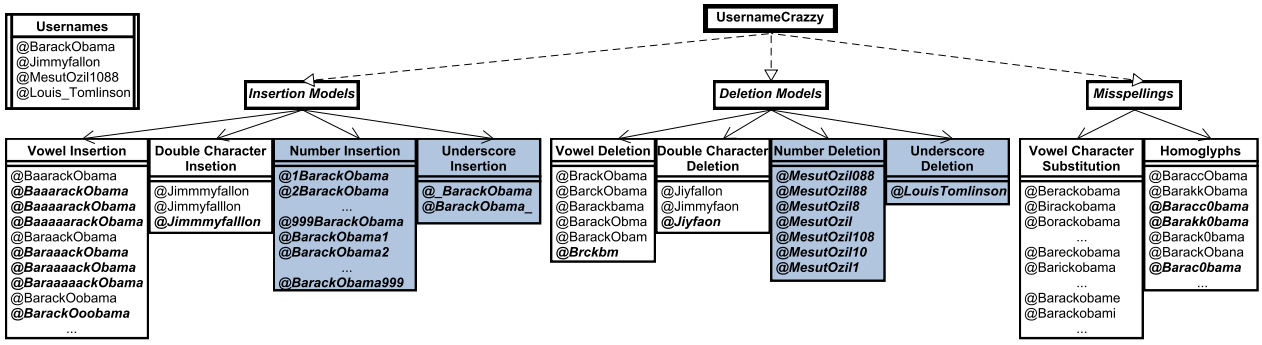

Figure 2: Taxonomy of UsernameCrazzy's generation models. The *blue-shaded* boxes indicate models which are not supported by existing tools. In *boldface* are username examples which cannot be generated by existing tools.

most. String generation models used in squatting methodologies can produce string variants that are significantly different from the seeded input but also among themselves. These can change the phonological properties of usernames, but we hypothesize that smaller changes can be more ng as they can result in similarly sounding and similarly looking usernames. To measure this, we compute the edit distance (ED) between a squatted user and its seeded one using a well–established string similarity metric [113].

3.2.4 ***MQ4.*** *Squatted Account Behavior.* To understand whether squatted accounts exhibit suspicious behavior we aim to identify i) how many accounts out of the generated ones have already been suspended, ii) how many accounts are bot accounts, iii) the behavior of squatted accounts that share similar profile picture and name.

Suspended Accounts Analysis. We use the Academic API v2 and we apply the `User Lookup` [106] method on every generated username. This call returns useful static information of an individual or a group of accounts, such as the *name*, *username*, and the *creation date*. If an account is suspended, we conclude that these accounts were involved in some form of malicious (e.g., impersonation) or unwanted (e.g., on) behavior [97]. As these accounts no longer exist on the network, *X* API does not return any output for a suspended username. To find the content of those profiles we instead used the *Wayback Machine* [46]; an online archive that maintains snapshots of *X*[3]. Two independent raters then analyze the recovered profiles. We based the content analysis on *X*'s prohibited activity [101] and classified the profiles as i) impersonation attempts; ii) bots engaged in automated or spamming activity; iii) accounts that explicitly mention they are fan/parody/fake version of the celebrity account; this kind of 'impersonation' is allowed by *X*'s policy [103]; iv) benign users, i.e. users who are not related to the celebrity account except for sharing a username variant and v) other accounts which do not fall in the above categories.

Active Accounts Analysis. We first run Botometer [78] for *every* generated variant to understand how many of the accounts are bots. Botometer supports two types of scores, '*English*' and '*Universal*' (language independent). The former takes into account 6 categories of features, while the latter omits 2 of them (content and sentiment features) which are English-specific. Note that the '*Universal*' score is not as accurate as the '*English*' [74, 38]. Consequently, during our experiments we try to work with English-related users as much as possible. We note that we examine only a subset of our users because Botometer does not return a score for locked accounts.

To gain a deeper understanding into the activity of the active profiles, we perform a manual analysis of the activity of 1,400 randomly selected squatted username accounts that have a profile picture with a face or an avatar. Based on previous works [4, 87], profile pictures with faces are more persuasive in terms of influencing user behavior, engagement or credibility compared to pictures with other objects. Two independent raters then analyse the behavior of the accounts. If an account had no suspicious behavior it was labeled as *benign*. If an account can contribute to on we term it *suspicious*; these were subdivided into the following: a) *impersonation*: an account that has no specific goal other than impersonate, e.g., this impersonation could be done for entertainment purposes or the creator of the account could be a fan, but does not explicitly mention so on the account [103]; b) *financial* accounts that use the generated on for a financial incentive, e.g. they try to sell products or gain followers; c) *political* aiming to use the on to influence their followers politically, e.g., such accounts may (re)post news or opinion articles; d) *news* accounts that spread (fake) news about the celebrity they impersonate, these accounts differ from the impersonation accounts as they have the specific goal of spreading news about the seed account holder; e) *harass*: accounts that directly harass (e.g., insult or aggressively mention or retweet) the celebrity they impersonate or other celebrities; and f) *other*: other accounts that did not fall into the other categories, e.g., accounts with little or unrelated activity.

Amplification of Confusion. Important to our analysis is to understand whether squatted accounts with malicious purposes exploit other similarities with the original account. To measure this we search for active squatted accounts with similar *profile names* and *images* to the original and analyze how many of those accounts are bots. We apply VGGFace2 [11], a popular image recognition model (see Section 5.1 and Algorithm 1 in Appendix C for details of the algorithm) on all squatted usernames [4] and then we compare, using the Levenshtein distance [113], their profiles names with their potential target. We consider profile names to be similar if either i) two names are equivalent, ii) an extra character is added as a

[3]The snapshots for which we get a '*Got an HTTP 302 response at crawl time*' error are excluded from the analysis.

[4]Profile images that do not depict a face are excluded.

prefix or suffix, iii) a substring of the original name appears as a substring in the squatted name (e.g., for 'Cristiano Ronaldo' we check whether 'Cristiano' or 'Ronaldo' appear in the squatted account's profile name). Details of the algorithm can be found in Appendix D, Algorithm 2. Out of those users which share similar image and/or name, we compute how many are bots using Botometer [78].

4 CHARACTERIZATION

4.1 Prevalence of Username Squatting.

We conducted our **MQ1** measurements on *X*, in October 2021 and found a total of 41,393 active username variants from the *Initial Seed* ($n = 97$) of popular accounts, or approximately 427 variants per username. Table 3 lists the 5 top original usernames with the most active username variants. We also found that 10,361 accounts with username variants were suspended, or 107 on average per original account. This is a good signal that in the past these accounts were malicious [97] which we further analyze in Section 4.4. Next, we examined how these suspended accounts are spread across different categories. We found that 7,940 accounts (66.3%) belong to the 'Entertainment' category, 1,175 accounts (9.8%) in 'Sport and Gaming', 1,104 accounts (9.2%) in 'Companies, Brands and Organizations', 709 accounts (5.9%) in 'Government', 582 accounts (4.8%) in 'Activists, Organizers and Other Influential Individuals' and 460 accounts (3.8%) in 'News Organizations and Journalists' category. In May 2022, we ran again the measurements and found 43,349 active and 11,384 suspended accounts while in November 2022 we found 43,994 active and 11,827 suspended accounts (see Table 3). Alarmingly, the increase on the number of the suspended users in every new snapshot indicates that this is an ongoing and growing problem. For the rest of the paper, we use the first snapshot. Thus, we conclude that celebrities and popular users in general are indeed a common target of username squatting attack, something that contrasts the insights of previous work which found 166 potential impersonators and only 3 of them were celebrity impersonation attempts [35] (see Sections 7 and 8).

Table 3: Top 5 original accounts with the most active username variants on *X*.

	Potential Squatted Usernames					
Original Accounts	**Generated Accounts**	**Gen. Accts. with ED 1-3**	**Active Accounts**	**Act. Accts. with ED 1-3**	**Suspended Accounts**	**Sus. Accts. with ED 1-3**
@kaka	5,899	1,380	2,145	441	224	96
@cristiano	17,687	2,301	1,714	639	148	63
@pink	11,955	1,348	1,555	381	162	75
@nasa	5,842	1,356	1,534	333	231	42
@selenagomez	9,925	2,270	1,346	306	177	97
Total (Oct. 2021)			41,546	37,965	10,208	9,762
Total (May 2022)	851,682	378,088	43,349	39,403	11,384	10,781
Total (Nov. 2022)			43,994	40,200	11,827	11,133

4.2 Online Confusion.

User Mentions. For **MQ2**, by applying the methodology described in Section 3.2.2 we collected 1.2 million tweets to measure the contribution of squatted accounts to online confusion. Tweets can either be *actual tweets*, *retweets*, or *replies* that mention at least one of the generated username variants. We found that out of the 864,369 actual tweets, 657,337/864,369 (76%) are likely *typo-mentions* and only 200,562/864,369 (23.2%) happen when the mentioner and mentionee are within one hop on the social graph (*purposeful–mentions*). We discard the rest of the tweets (6,470 or 0.8%) because the API did not return information about the friendship of the users.

We further analyze the 657,337 *typo-mentions* and found that 437,550 tweets (66.6%) belong to the 'Entertainment' category, 96,720 tweets (14.7%) in 'Sport and Gaming', 57,126 tweets (8.7%) in 'Companies, Brands and Organizations', 33.220 tweets (5.1%) in 'News Organizations and Journalists' category, 25,206 tweets (3.8%) in 'Government' and 7,515 tweets (1.1%) in 'Activists, Organizers and Other Influential Individuals'. We observe that typo-mentions are prevalent in all categories but most are found in Entertainment accounts as expected since our dataset contains more such accounts. Typo-mentions are frequent which is worrisome since they can drive more traffic to the suspicious account.

User Search Recommendations. In order to identify the factors that can lead to the increased popularity of potential squatted accounts, we take a closer look at the suggestion algorithm of *X*. We expect that when a user searches for a celebrity using the search bar the targeted profile should be returned within the first 10 options due to its popularity. Using the methodology from Section 3.2.2, out of the 929 requests (the total number of the characters of our *Initial Seed*) we find that only 166 and 178 times all the original users together appear in the first 10 and 100 recommendations, respectively. Alarmingly, only when the full original usernames were inserted in the search bar, the official profiles were returned as the first 15 options (see Table 4). We observed that there exist instances where the squatted username appears *before* the original account. Clearly, this is problematic as it can exacerbate the problem of online confusion. More precisely, 1,205 of active squatted accounts appear in the first 1,000 recommendations, 705 of them appear in the top 500 recommendations, 170 in the top 100 and 26 appear in the top 10 recommendations while searching for our seed accounts (see Figure 3). This indicates that the popularity of squatted usernames can be amplified through the search algorithm of the platform itself. Moreover, we manually check the 170 squatted users appearing in the top 100 recommendations and marked 10% as impersonators and 38.8% as malicious. This shows alarmingly that search recommendations can not only amplify confusion but can also be leveraged by malicious actors.

Table 4: Original accounts within the first 10 and 100 search recommendation results from a total of 929 requests.

	Original Accounts - *X*'s Recommendation System	
Position	**# of Times Original Accounts Recommended**	**Observations**
Top 10	166	108 times the original accounts returned as first recommendation but only when their full name was inserted in the search bar (ED = 0)
Top 100	178	132 times the original accounts returned in position 11 to 15 but only when their full name was inserted in the search bar (ED = 0)

4.3 Squatted Username Characteristics.

For **MQ3**, we aim to analyze the characteristics of squatted usernames to identify which ones are more prevalent or contribute to confusion the most. To do that we use the edit distance measure (see Section 3). The highest number of suspended and active accounts are squatted usernames where the *ED* is between 1-3 characters (see

Figure 8 in Appendix F). Looking into the high numbers for $ED = 3$, we found that in numerous cases, username variants consist of an addition of *three* digits before or after the seed username.

Making 1-3 mistakes when typing a word is reasonable [19, 34]. Figure 4 depicts that i) indeed users often make 1-3 mistakes in their mentions and ii) most of the times there is no relationship between the mentioner and the mentionee. The number of mistakes in actual tweets is more informative than in replies or retweets, as the latter can simply propagate the same typo mistakes. Figure 3 illustrates the effects of the edit distance on X's search recommendations. Out of the squatted accounts suggested by the search algorithm, again we note that those with $ED = 1$ are most common. Interestingly, 24 squatted users appear within the top 10, 164 within the top 100 and 686 within the top 500 of the search recommendations. From these results we can conclude that an adversary can identify specific patterns of typo mistakes and claim a username that is close to the original. Our results indicate that typo-mentions and the search algorithm can be exploited to increase traffic to squatted accounts.

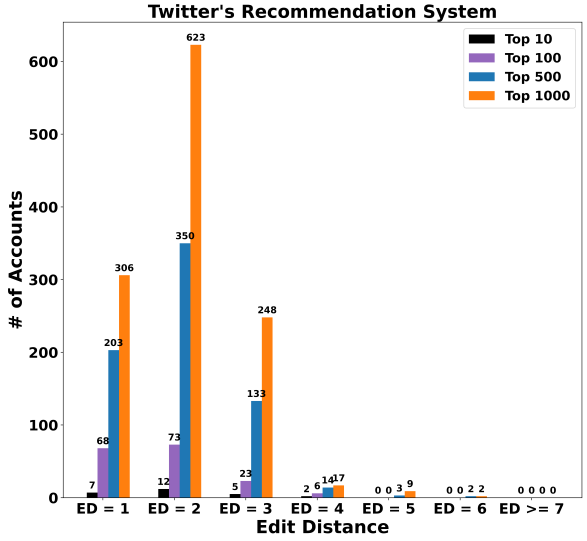

Figure 3: Squatted accounts in the top search recommendations grouped by edit distance.

4.4 Squatted Account Behavior.

For **MQ4**, we aim to analyze the activity of both suspended and active accounts. For this we manually analysed the behavior of 135 suspended accounts using the Wayback Machine, we classified the behavior of 540 suspicious and active accounts and we applied Botometer to all active accounts.

Suspended Accounts Analysis. By applying the methodology described on Section 3.2.4 we found 10, 208 suspended accounts. *Wayback Machine* [46] returned valid snapshots for 135 of those accounts . Then, two independent raters went through all the available snapshots and they found (see Table 5): 52 impersonation attempts of the original account, 12 accounts posting e.g., spam links or product promotions, 6 human accounts that explicitly mention they are fan or parody accounts, 15 profiles of humans with no connection to the original account, 45 other accounts. Other include private profiles or accounts with no activity. Also, for 5 accounts there

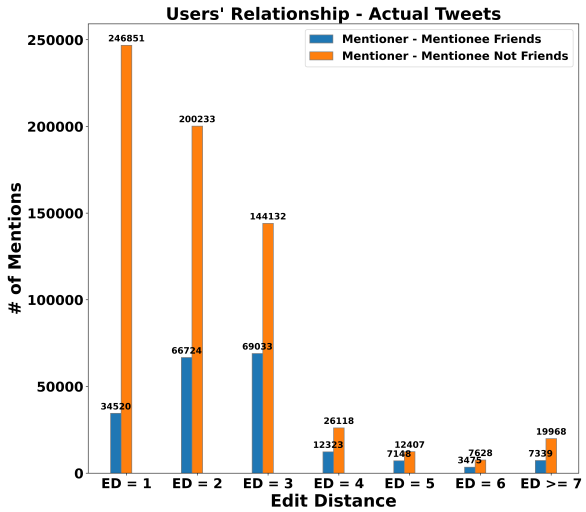

Figure 4: Number of actual tweets that mention at least one username variant grouped by the variants' edit distance with their original accounts.

was a disagreement between the raters so we do not report their results. This analysis provides a first validation of our idea that a suspension was the consequence of malicious content creation.

Table 5: Manual classification of suspended accounts.

Suspended Accounts (135 users)				
Impersonations	**Bots**	**Fans**	**Humans**	**Other**
52 (38.5%)	12 (8.8%)	6 (4.4%)	15 (11.1%)	45 (33.3%)

Active Accounts Analysis. Previous works [5, 23] used Botometer as ground truth for measuring the bot activity of accounts. To validate our hypothesis that many of the squatted usernames are indeed malicious and/or bots, we applied Botometer to all the generated usernames that are still *active* on X. Botometer uses *CAP* score, a probability which indicates that a profile with this score or greater is controlled by a software (i.e., is a bot). We set *CAP* score to 0.95, which is a conservative value expected to yield only 5% false positives [78]. Our analysis shows that on our dataset this value can be around 11%. Out of 41, 546 variants Botometer returned a score for 25, 089 since the other accounts are private profiles. 9, 702 (38.7%) profiles were classified as automated users. The automated accounts are further categorized as different kinds of bots and the largest category was *fake followers*. We observe that a large number of the squatted users are malicious bots, indicating also the effectiveness of UsernameCrazy.

We next performed a manual analysis (see Section 3.2.1) to better understand the *type* of activity the squatted accounts are involved in. Out of the 41, 546 active profiles we randomly selected 1, 400 users who have a face or an avatar in their profile picture. Two raters agreed beforehand on the rules they will follow, discussing about the characteristics of impersonators, spam/bot and benign users. Accounts that had no similarities with the seed account or explicitly say they were fan or parody profiles were labelled as *benign* [103], while accounts that were sharing same features with their original accounts were labelled as *suspicious*. Also, users

sharing a number of features[5] as their seed account but with limited or no activity were labelled as *suspicious*. The process allowed us to identify 838 accounts as *benign*, 540 as *suspicious* (defined as an account that can contribute to confusion) username squatting attempts and 22 accounts were left as 'difficult to declare'. We examined the inter-rater agreement using Cohen'a Kappa [66] and found $\kappa = 0.92$ and $\kappa = 0.89$ for the benign and suspicious users, respectively, which indicates *near perfect* agreement. Then, the two manual raters further categorized the suspicious accounts' behavior as *impersonation*, *financial*, *political*, *news* and *harass*. The full results for the suspicious accounts are presented in Table 6. We conclude that a large amount of active accounts are involved in some form of malicious or confusing behavior.

Table 6: Manual classification of the active malicious accounts used in dataset.

Active Malicious Accounts (540 users)					
Impersonations	**Financial**	**Political**	**News**	**Harass**	**Other**[6]
115 (21.3%)	56 (10.3%)	7 (1.3%)	15 (2.6%)	4 (0.7%)	343 (63.5%)

Analysing Confusion Amplifiers. Using the method presented in Section 3.2.4 we analyze how many bot accounts share similar features (here profile name/image) with the original one. Figure 5 depicts that indeed a large number of users have a similar name/image with their target; this suggests that these profiles likely impersonate the original owner to drive traffic to their account and increase the impact of their malicious activity.

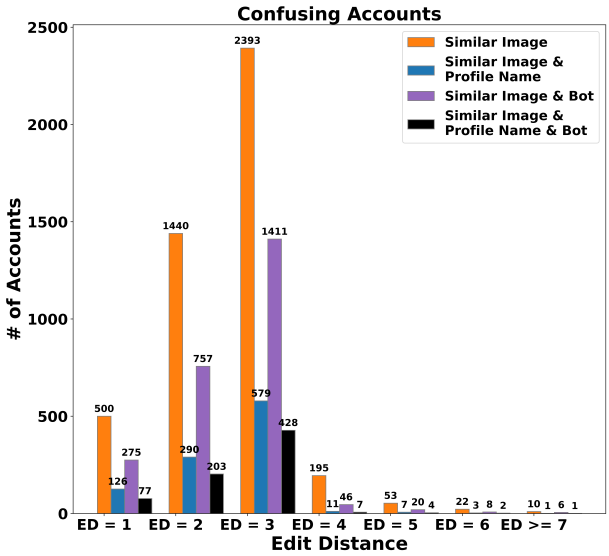

Figure 5: Classification of the confusing accounts.

5 SQUAD DESIGN OVERVIEW

Our measurement study revealed that username squatting is an important issue on X as a significant subset of these accounts are bots or impersonations involved in malicious behavior. On the other hand, a large number of squatted users are benign (e.g., fan/parody users) and we need an effective way to make that distinction. This raised the following question: *Can we design an effective and efficient framework for detecting suspicious username squatting attempts?*

Toward this end, we developed a novel methodology embodied in an end-to-end username **SQUA**tting **D**etection framework (SQUAD) to facilitate the detection of potentially malicious confusion attempts against popular accounts. To do that, SQUAD uses UsernameCrazy in combination with a classifier to output accounts that are confusing and require further analysis.

Figure 6 illustrates the overall architecture of the framework. First, the seed account for which one wants to find confusing versions is selected. This is used as input in UsernameCrazy for the generation of squatted usernames (see Section 3). These are fed into a *filtering* component which identifies active accounts corresponding to the squatted usernames [7]. The active accounts and the *Initial Seed* account, are fed into a *feature extraction* component which extracts certain account features. These are then passed into the *classification* component. This outputs pairs of <*initial, squatted*> accounts where the squatted account is sufficiently confusing and requires further analysis. The username generation and filtering components were introduced during our measurement study. Next, we focus on the feature extraction and classification.

5.1 Feature Extraction and Selection.

Initial Feature Set. We start with an overview of the account features our classifier used. We manually select the features that could play a role in the account being confusing. This selection was based on our observations and findings from prior works identifying impersonation attempts and bot accounts [26, 70, 78, 35].

We selected a number of features relevant to comparing the similarity of two accounts in terms of their image, bio, profile name, username and URL. For the *Image Similarity Score*, we use an image recognition model, VGGFace2 [11], to identify whether the squatted account's profile image is of the same person as the seed account's. It makes use of a convolutional neural network, known as SE-ResNet-50 [42, 11]. To ensure accuracy of the model on our dataset we first performed a test on a manually labeled subset of 128 images[8]; this resulted in a 85.93% test accuracy for the cutoff threshold 0.65 (see Table 10 in Appendix B for the full results). Note that all images with a higher score are classified as '*no similarity*'. The test accuracy is lower than previously reported (90.8%) [11] but still high enough to justify using this model for our prototype implementation. We also use *Binary Image Similarity* which returns 0 and 1 if the similarity score is below or above the selected threshold respectively.

For *bio similarity* we use the *Jaccard Distance [24]* which is a similarity metric between two sets. The similarity index ranges from 0 to 1. The closer to 1, the more similar the bios. Even though *Jaccard similarity* does not have a notion of semantics, we found it to perform comparably with calculating the distance between the embeddings of bio statements, where we calculated 200-dimensional embeddings [73] pre-trained on Tweets. We also convert any special character (e.g., emojis) in to word format. For profile and username

[5] Profile name or picture, username.
[6] Accounts with little or unrelated activity.
[7] The filtering also identifies suspended, deleted and non-existing accounts which might be useful for further analyses.
[8] We removed accounts that did not represent a human (e.g., companies and teams) or that portray groups of people, like bands. As a result, we are left with 64 accounts. For each of these accounts we pick 1 random generated variant and for every variant we select 2 random profile pictures, resulting in 128 profile pictures in total.

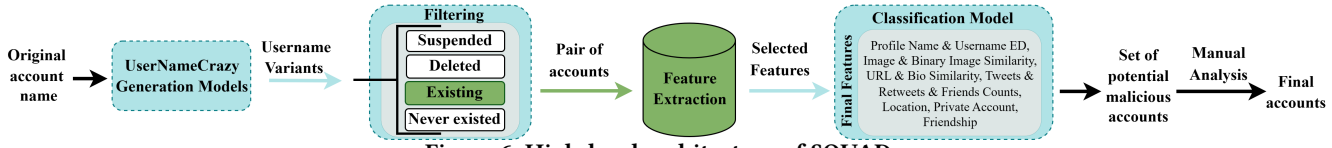

Figure 6: High-level architecture of SQUAD.

similarities we compute the Levenshtein distance [113]. To calculate URL similarity we collect the textual representation of a profile's 'website' field for both users. We first check whether the URLs are exactly the same or the squatted account uses a URL that includes the seed name as a substring of it. If any of these is true, the result is '1'. If the users use completely different URLs the result is '0'.

We also used several count-based features such as *Friends Count*, *Followers Count*, *Tweet Count*, and *Retweet Count*, and the features *Private*, *Friendship*, *Location*, and *Tweet Sentiment*. *Private*, indicates whether an account is private. The *Friendship* is a binary feature which shows if a squatted account follows the seed account. In terms of *Location*, for every account, we collect the textual representation of the mentioned location, like the city name. We then check whether a squatted account shares the same location with the seed one. If yes, the returned results is '1' (this is applicable also if both locations are empty). Otherwise, the result is '0'. Lastly, for analyzing tweet sentiment we fetch the 500 most recent tweets of the squatted account. We use NLTK [7] for preprocessing the text to remove the stop-words, user-mentions and punctuation marks, and then we apply TextBlob [62], a sentiment method for classifying the tweets as *neutral*, *positive* and *negative*. For the accounts with no tweets, the score of every category is '0'. Since in almost 95% of the cases the returned result was 'neutral', we discard this feature. Note also that we normalize all the data between 0 and 1 based on the min-max normalization process [112].

Feature Selection. The initial set of features showed success in relevant classification tasks but not all of them are necessarily important for our task. In fact, redundant features can decrease the performance of the classifier by introducing noise [10, 77]. To remove such features we select the optimal feature set via the *Recursive Feature Elimination* (RFE) with Cross-Validation process [72, 14], optimizing for accuracy. Our final feature set includes the *Profile Name and Username Edit Distance*, *Image Similarity Score*, *Binary Image Similarity*, *Friendship*, *Friends and Tweet Count*, *Bio and URL Similarity*, *Location*, *Retweets Count* and *Private Account* features.

5.2 Generating the Dataset.

For each <seed, squatted> account pair, the extracted features will be used in a binary classification whose output is the label *benign* or *suspicious*, where suspicious refers to an account that can contribute to confusion. The goal of a successful classifier is to have a low false positive (a positive being a suspicious account) rate, so that when the tool is used in combination with manual analysis it allows to find malicious activity on the OSN as efficiently as possible.

A main challenge in our framework's classification modeling is that there are no publicly available datasets of impersonation attempts which could use to train and evaluate candidate models. Existing ones, such as [49, 50, 52], are either datasets that consist of bots, or fake accounts which are not trying to imitate a specific profile. Our definition of confusion requires a different dataset as it encompasses any account that shares a sufficient number of features with the seed account including bots, impersonations and other accounts involved in malicious activity. Therefore, we created our own dataset by manually labelling a subset of the data records we collected for our measurement study. We use the same 1378 users which two raters manually and independently label as *benign* or *suspicious* (see Section 4.4). To fix the imbalance of our dataset, we produce more examples from the minority class by applying the Synthetic Minority Over-sampling Technique (SMOTE) [13]. We then use our labeled accounts to train and evaluate 7 popular binary classification models which our prototype implementation of SQUAD supports. In Section 6 we compare their performance.

6 EVALUATION

The goal of SQUAD is to speed up the process of finding malicious activity on the network by identifying confusing accounts that require further manual analysis. It uses UsernameCrazy to avoid searching the entire social graph by *reducing* the search domain. The filtering component helps identify from that username search domain which variants exist on the network. Here, we first determine *the effectiveness and efficiency of UsernameCrazy in identifying squatted accounts and how it compares with previous string squatting tools* (Section 6.1). Then we analyze *the accuracy of SQUAD in classifying confusion accounts* (Section 6.2).

6.1 UsernameCrazy Performance.

Effectiveness. First we evaluate the ability of UsernameCrazy to produce *relevant* username variants and compare it with prior tools which leverage squatting techniques for generating complex strings, namely *URLCrazy* [108] and *AppCrazy* [43]. We use the 10 most popular users of our '*Initial Seed*' as input to the three generation tools and then apply our filtering component. We report the number of username variants produced, the number of those that actually exist on the network, the number of produced username variants that are already suspended by the network and the number of the existing impersonation attempts found based on our manual analysis. Table 7 summarizes our results and a full break down per user can be found on Table 9 in the Appendix A.

We observe that UsernameCrazy exhibits better username generation coverage compared to existing methods. It generated $O(10^4)$ usernames compared to $O(10^3)$ and $O(10^2)$ from *URLCrazy* and *AppCrazy*. This is a result of UsernameCrazy's enhancements with new models, *model self-repetition*, and *model stacking*. We also observe that compared to both prior methods UsernameCrazy identified one order of magnitude more variants that exist on the network and that are suspended. This showcases UsernameCrazy's effectiveness in producing relevant string variations. There are several reasons for this. Firstly, OSNs have constraints on the number and type of characters a username can have. Prior tools do not take that into account and use generation methods (such as bit flipping or string rearrangement) which produce a large number of usernames that

Table 7: Competency of *URLCrazy*, *AppCrazy* and *UsernameCrazy*.

	URLCrazy				AppCrazy				UsernameCrazy			
Usernames	# of Generated Usernames	# of Active Squatted Usernames	# of Suspended Squatted Usernames	# of Active Impersonators	# of Generated Usernames	# of Active Squatted Usernames	# of Suspended Squatted Usernames	# of Active Impersonators	# of Generated Usernames	# of Active Squatted Usernames	# of Suspended Squatted Usernames	# of Active Impersonators
Total (10 users)	**1,274**	**215 (16.87%)**	**49 (3.84%)**	**10**	**436**	**293 (67.20%)**	**73 (16.74%)**	**17**	**96,968**	**6,629 (6.83%)**	**1,901 (1.96%)**	**88**

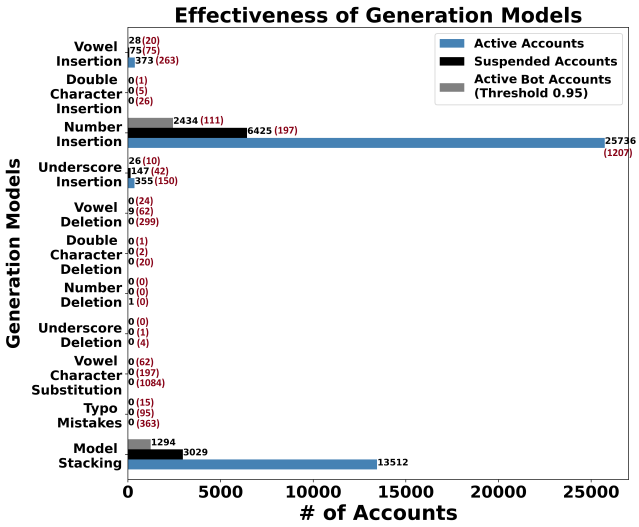

Figure 7: The effectiveness of a) each generation model with *self-repetition* and b) *model stacking*. In *parenthesis* is the effectiveness of the primitive models—not shown on the bars.

are invalid to begin with. Further, other tools produce strings that include a top-level domain either at the head or tail of the name making them again automatically incompatible.

Note that the ratio of existing or suspended usernames over all generated usernames is low. This is because a lot of the usernames are valid but have not been used yet. Nonetheless, it is important to be able to produce them to discover future instances of squatting username attempts.

Effectiveness of Generation Models. Next, we want to better understand which of UsernameCrazy's generation models are the most important. We repeat our effectiveness analysis for the *primitive models*, the generation models with *self-repetition* and the *model stacking* of UsernameCrazy for all the usernames of the *Initial Seed*. We compare the number of produced username variants that actually exist on the network or have already been suspended by the platform. We also measure how many active users found are marked as potential bots by Botometer [78] as this is a further indication of the ability of UsernameCrazy to produce relevant usernames. We set Botometer's detection threshold to 0.95 (see Section 4.4). Figure 7 summarizes our results. We observe that a very large number of the generated variants have been either already suspended or declared as potential bots from Botometer. This demonstrates that our username generation models can reveal numerous squatted accounts that are potentially malicious. Moreover, we observe that the number insertion method is the most successful in identifying active accounts and that the majority of such accounts are either suspended or marked as bots. This reveals that a common username squatting tactic on OSNs is to add a digit as a prefix or as a suffix on a popular account's username. This strategy is simple to both conceptualize and implement. Considering also our measurements on edit distance (Section 4.3) we observe that adding between 1–3 digits on a target username seems to be the most common strategy.

UsernameCrazy Runtime Efficiency. Since UsernameCrazy can generate thousands of candidate usernames, it is important to do that efficiently. To measure UsernameCrazy's runtime performance, we provided as input to UsernameCrazy the *Initial Seed* (i.e., the usernames of the 97 most popular users for X). We set the maximum character count of a variant to be generated to be 15 (which is also the maximum allowed for X usernames) [107]. UsernameCrazy ran on commodity hardware (a laptop with an Intel Core i5-8265U processor with 8 GB of RAM). UsernameCrazy produced 851, 682 valid usernames in 7.6 seconds, which verifies that it can be a practical component in an end-to-end discovery of squatted accounts.

6.2 Classification Performance.

Training and Evaluation Metrics. UsernameCrazy produces relevant usernames but not all of them are used with malicious intent. Our classification component aims to take pairs of <seed, squatted> accounts and decide whether the squatted version is suspicious. To evaluate the classification performance of the model we split our dataset to 70% (965 users) and 30% (413 users) for training and testing, respectively. Then, we evaluate 7 of the most popular classification algorithms that have been shown to perform well on binary classification tasks: Random Forest [40], Naive Bayes [63], Logistic Regression [17], K-Nearest Neighbor (KNN) [67], Support Machine Vector (SVM) [16], Decision Tree [115], Neural Network (NN) [37]; all using the default parameters. *Precision*, *Recall* and *F1-Score* [22] have been used as evaluation metrics. We found that Random Forest performed the best in all metrics. To improve its performance we fine tune its parameters by applying '*Hyperparameter Tuning* [31]' on all parameters, namely '*n_estimators, min_samples_split, min_samples_leaf, max_features, max_depth* and *bootstrap*'.

Overall Performance. Random Forest performs better than the other classifiers, with 94% and 94.5% average precision and recall, respectively. Table 12 in Appendix G summarizes the classification performance results of all models. While not shown in the table, the accuracy of the Random Forest is 94.44%. The model correctly classifies 154/162 accounts as malicious. We also compute the '*Mean AUC*' score for all classification models and illustrate the trade-off between the false positive and true positive rates in Figure 9 in Appendix G. We observe that there is an 99% chance that Random Forest will correctly distinguish negative and positive classes, outperforming the other models. The number of false positives plays a crucial role in eliminating the manual analysis effort for identifying malicious attempts. After a closer inspection we find that our model

returns 8 false negative cases, but even though it misses a few cases we already demonstrated that our approach is vastly superior to existing squatting generation techniques while it is more practical than expensive deep graph traversals. We also find 15 false positives which shows that SQUAD keeps the number of false alarms low.

Improving SQUAD. SQUAD's dataset was limited to a relatively small number of manually labelled users. One potential improvement could be to increase the size of the labeled dataset.

We also investigated the false positive (FP) and false negative (FN) cases. For the 15 benign users SQUAD misclassified (FP) we observed that a) all users have small *username* edit distance (≤ 3), b) 7/15 accounts have small *profile name* edit distance (≤ 3), c) 9/15 profiles have similar image with the target and VGGFace2 returns a score lower than the threshold (marked as similar). These explain why SQUAD predicted them as suspicious. Looking closer at the latter set, we observed that (d) 6/9 use the word 'fan' or 'parody' in their bio and 4/6 of these exhibited a high bio similarity (> 0.6). We leveraged these and introduced a post-filter on SQUAD which uses a keyword search for 'fan'/'parody' in account bios and negates the suspicious label if it finds a match. This reduced our false positives by almost 30%. We leave integrating more complex methods based on word/sentence embeddings on the bios for future work.

For the 8 malicious accounts SQUAD missed (FN) we made the following observations: a) all accounts have small *username* edit distance (≤ 3), b) half of the accounts have small *profile name* edit distance (≤ 3), c) 6/8 accounts have similar image with the target but VGGFace2 returns a score greater than the threshold and d) 4 of the latter accounts have also very low bio similarity (< 0.10). As a result, the combination of the misclassification of the image and the low bio similarity lead to a wrong classification. Thus, SQUAD can improve as VGGFace2 or image recognition improves—its modular design allows for simply exchanging the face recognition module with another one. Also, SQUAD only supports image similarity of faces and avatars. Hence, it can miss cases where an impersonator uses a different subject than a face in its profile photo, but as shown in prior work such strategies are less effective [4, 87]. Lastly, SQUAD can benefit from integrating more advanced models for characterizing users' bios, activity and tweets.

7 APPLYING SQUAD

Applying SQUAD to Non-Popular Accounts. Using the CrowdFlower AI gender predictor dataset [105] we randomly select 15 non-private profiles with less than 200 followers (98% of *X* users have less than 400 followers [83]). Users without a face in their profile picture are disregarded from the selection. SQUAD returns 159 active accounts where 61 have a face in their picture. Out of the latter, 0 (0%) classified as *suspicious* and 61 (100%) as *benign* accounts by SQUAD. After manual verification, we find only 1 *false negative* case. The results contradict the insights of [35], where any account on *X* can be a victim of impersonation. We summarize our findings in Table 11 (see Appendix E).

Applying SQUAD to Celebrities. We apply SQUAD to 21 [8] users of our *Initial Seed* (see Section 3.1). SQUAD returns 3,295 active accounts where 1,137 have a face in their profile picture. Out of the latter, 625 (55%) classified as *suspicious* and 512 (45%) as *benign* accounts by SQUAD. Alarmingly, 297 (47.5%) users from the suspicious set were generated by the *number insertion* strategy, signifying the model's malicious squatting intent (see Section 6.1).

Analysing the Behavior of Suspicious Profiles. To find if the suspicious variants are engaged in malicious activities, we analysed the content of their tweets. Using the full-archive search endpoint (API v2 [104], see Section 3.2.2) we fetch their 500 most recent actual tweets, resulting in 8,516 tweets where 1,476 (17.3%) include a URL and 5,056 (59.3%) are *English* tweets. We manually observed two trends in the tweets of malicious users which we further measured. Firstly, they share URLs which sometimes look suspicious and secondly they try to grow their follower base. In terms of URLs in tweets, we first analyze their integrity. Since none of the URLs are present in the VirusTotal database [81] we submit them for scanning. Alarmingly, we discover: i) 24 (1.63%) URLs were marked as malicious by 1-3 antivirus systems, ii) 533 (36.1%) URLs fail to use *HTTPS*, and iii) 16 (1.08%) URLs automatically obtained a TLS certificate from *Let's Encrypt* Certificate Authority [61], a service that has faced criticism for its decision to abstain from implementing any form of security checks prior to granting certificates to domain owners [2]. We also measured follower base growth attempts in the English tweets using a simple keyword search approach, looking for the phrase *follow me* (with slight variations acceptable). We found 266 (5.3%) such cases, being posted by 78 distinct users.

Username Squatting across Platforms. Username squatting can be exploited on other OSNs that use usernames as unique identifiers to search for and interact with users, such as Instagram and TikTok. To illustrate the extent of the problem on other platforms we would ideally have to (a) adapt UsernameCrazy generation models to the username constraints of each platform, (b) collect existing variants on each platform and, (c) classify each account. The former is easy to do and can be efficiently executed. (b) on the other hand is harder especially since there are no publicly available APIs to allow us to automatically crawl Instagram and TikTok. Hence, this would require developing specialized web crawlers for each case. Lastly, (c) is also hard as it would entail manual analysis on either all or a relatively large subset of the existing variants. Instead, we design a more targeted experiment which can quickly allow us to collect preliminary indicators of the extent of the problem on other OSNs.

We observe that popular profiles maintain accounts on other OSNs with the same username. Hence, if other OSNs are plagued with similar issues, then it is likely that variants that were marked as malicious on *X* could appear on these OSNs too. To examine this, we select the top 10 accounts of our seed and manually check how many exist with the same username across Instagram and TikTok; this resulted in 5 profiles. For each profile we use the squatted usernames, previously generated by UsernameCrazy, and find how many still exist on *X*. Then, for every active variant we manually select the ones that are likely impersonators and examine whether they a) exist with the same username, b) have a similar image and c) perform potential malicious activity on other OSNs. We call '*Level 1*' the users who cover only the first property, '*Level 2*' the profiles which cover the first property plus any of the other two and '*Level 3*' the accounts that cover all the aforementioned properties.

Table 8 summarises our findings. The most common pattern of the active squatted accounts across OSNs remains the *Number*

Insertion pattern. Lastly, almost half of the squatted accounts that impersonate on *X* exist on other OSNs and have common properties.

Table 8: Similar impersonation attempts across different Social Networks.

Impersonations Across OSNs				
Users	***X* - Active Impersonations**	**Category**	**Instagram**	**TikTok**
@katyperry	7	Level 3	1/7	0/7
		Level 2	4/7	4/7
@justinbieber	20	Level 3	3/20	4/20
		Level 2	7/20	10/20
@cristiano	1	Level 3	0/1	0/1
		Level 2	1/1	1/1
@ladygaga	10	Level 3	0/10	1/10
		Level 2	3/10	7/10
@kimkardashian	5	Level 3	2/5	0/5
		Level 2	3/5	0/5
Total	**42**	**Level 3**	**6/42**	**5/42**
		Level 2	**18/42**	**22/42**
		Level 1	**27/42**	**28/42**

8 RELATED WORK

Fake Accounts on OSNs. Prior work cloned OSN profiles without proposing ways to detect them [6] and studied the prevalence of profile name reuse on OSNs [65, 64]. Researchers have presented ways for detecting Sybil attacks using e.g., visual profile similarity [118], string matches and users' relationship measurements to identify whether one's account has been a victim of identity theft [55], the network relationship and attribute similarity [48], honest networks to reveal dishonest nodes via an inference engine [20]. In contrast, SQUAD combines username squatting techniques and static profile features for malicious classification.

Impersonation on OSNs. Several works analyzed posts [120] and post reactions [119] to identify impersonation of celebrities on Instagram. Closest to our work, Goga et al. [35] detect whether a pair of accounts on *X* that portray the same person is an impersonation attack or whether one identity owns both accounts (avata-avatar pair). The proposed technique requires manual selection of similar profiles and relies on expensive social graph traversals. It also found that impersonation is not a prevalent threat for celebrities. We show the opposite. Notably, we are the first to demonstrate that username squatting is a central strategy to such attacks and use it to design an efficient detection tool. Also, Goga et al. argue that an avatar-avatar pair is not considered dangerous. However, most of our malicious cases (see Table 6) belong to this category (521/540 profiles) making these users an overlooked problem. Finally, SQUAD does not rely on the following assumptions: a) an impersonator should have a connection with its target (e.g., friends, commenting on tweets) and b) impersonators are in the neighborhood of impersonators. SQUAD can detect impersonators anywhere in the social graph.

Spam and Abusive Accounts on OSNs. Zheng et al. [123] create passive honey-profiles, log the friend requests/messages and find anomalous behavior. Yang et al. [117] use honeypots to identify spammers, extract common evasion tactics based on their behavior, evaluate proposed detection techniques [59, 82, 110] and develop a system to detect spammers on *X*. Others [36, 60, 89] focus on developing suspicious URL detection systems. Xu et al. [116] propose a heavyweight ML framework which leverages features based on the direct/indirect neighbor properties to find users who violate the policies of an OSN, including bots/spammers/fake users. In contrast, the features used in SQUAD are easy and computationally cheap to collect and were based on analyzing specifically impersonators.

Bot Accounts on OSNs. Previous studies [3, 12, 79, 121, 32] have focused on the behavior patterns of bot accounts, concluding that they confuse users and pose social, financial and political hazards. Besides, Oentaryo et al. [70] focus on benign bots; tools also exist for detecting misinformation [44]. In our work we use Botometer [78] to classify existing squatted accounts and understand their behavior. However, Botometer uses the characteristics of bots [101], and while *botness* is a good signal it is not sufficient for detecting impersonators. Instead, we believe that the above works deviate from the content of our study, rendering direct comparisons inappropriate.

Squatting Techniques. Squatting issues have been studied in different domains [85, 68, 54, 43, 56, 18, 122, 84], exploring how they trick users into illegal activities. Panagiotis et al. [54] focus on combosquatting which lacks a generative model. An adversary can postfix/prefix any word constrained solely by the character limits [107], resulting in an exponential increase in the number of users that need to be searched within a graph. Similar to UsernameCrazy, tools also exist for generating string variations given a seed string [108, 43, 28] but they suffer from several limitations . To the best of our knowledge we are the first to a) develop a string generation tool compatible with OSN usernames, b) apply squatting methods to discover and characterize squatted usernames on OSNs and c) study how adversaries can adapt them to create confusion.

9 CONCLUSION

Our study is the first to characterize *username squatting* on OSNs. We showed that *username squatting* can be used for confusion and impersonation. We found hundreds of thousands of *typo-mentions* mentioning squatted versions of popular accounts with no apparent mentioner–mentionee relationship, and demonstrated how the search recommendation algorithm of a popular OSN can unwittingly amplify online confusion. We discovered that thousands of squatted usernames are already suspended by *X* while tens of thousands of *active* squatted accounts are likely bots. While not all username variants are malicious, we show that *username squatting* can be a useful signal for identifying such accounts. We designed SQUAD, to aid detection of malicious accounts on OSNs. SQUAD incorporates new techniques for username squatting along with a similarity feature extractor and classification components. SQUAD can generate tens of thousands of relevant squatted usernames of a given account in a few seconds, and can achieve 94% F1-score in detecting suspicious squatted accounts when trained on a small dataset. SQUAD can be used by account owners to identify attempts to impersonate them, by regulating authorities such as the Federal Trade Commission (FTC, US) [30] or the Information Commissioner's Office (ICO, UK) [45], and the OSN platforms themselves who want to take measures to limit the negative impact impersonation and confusion can have on the community. Lastly, we revealed that username squatting targets primarily popular accounts and 36% of the suspicious accounts' tweets share insecure URLs and 5% of the tweets try to grow their or others' follower base.

REFERENCES

[1] 2022. 2022 strengthened code of practice on disinformation. https://digital-strategy.ec.europa.eu/en/library/2022-strengthened-code-practice-disinformation. Accessed: 2022-12-01. (2022).

[2] Josh Aas. 2015. Let's encrypt: the ca's role in fighting phishing and malware. Accessed: 2023-06-01. https://letsencrypt.org/2015/10/29/phishing-and-malware.html.

[3] Flora Amato, Aniello Castiglione, Aniello De Santo, Vincenzo Moscato, Antonio Picariello, Fabio Persia, and Giancarlo Sperlí. 2018. Recognizing human behaviours in online social networks. *Comput. Secur.*, 74, 355–370.

[4] Saeideh Bakhshi, David A Shamma, and Eric Gilbert. 2014. Faces engage us: photos with faces attract more likes and comments on instagram. In *Proceedings of the SIGCHI conference on human factors in computing systems*, 965–974.

[5] Christoph Besel, Juan Echeverria, and Shi Zhou. 2018. Full cycle analysis of a large-scale botnet attack on twitter. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 170–177. DOI: 10.1109/ASONAM.2018.8508708.

[6] Leyla Bilge, Thorsten Strufe, Davide Balzarotti, and Engin Kirda. 2009. All your contacts are belong to us: automated identity theft attacks on social networks. In (Jan. 2009), 551–560. DOI: 10.1145/1526709.1526784.

[7] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

[8] Social Blade. [n. d.] Top 100 most followed twitter accounts. https://socialblade.com/twitter/top/100. Accessed: 2023-12-01. ().

[9] Thomas Bohm. 2014. Letter and symbol misrecognition in highly legible typefaces for general, children, dyslexic, visually impaired and ageing readers. *Information Design Journal*, 21, (Dec. 2014). DOI: 10.1075/idj.21.1.05boh.

[10] Boseon Byeon and Khaled Rasheed. 2008. Simultaneously removing noise and selecting relevant features for high dimensional noisy data. In *2008 Seventh International Conference on Machine Learning and Applications*, 147–152. DOI: 10.1109/ICMLA.2008.87.

[11] Qiong Cao, Li Shen, Weidi Xie, Omkar M. Parkhi, and Andrew Zisserman. 2018. Vggface2: a dataset for recognising faces across pose and age. In *2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)*, 67–74. DOI: 10.1109/FG.2018.00020.

[12] Nikan Chavoshi, Hossein Hamooni, and Abdullah Mueen. 2017. Temporal patterns in bot activities. In (WWW '17 Companion). International World Wide Web Conferences Steering Committee, Perth, Australia, 1601–1606. ISBN: 9781450349147. DOI: 10.1145/3041021.3051114.

[13] Nitesh Chawla, Kevin Bowyer, Lawrence Hall, and W. Kegelmeyer. 2002. Smote: synthetic minority over-sampling technique. *J. Artif. Intell. Res. (JAIR)*, 16, (June 2002), 321–357. DOI: 10.1613/jair.953.

[14] Xue-wen Chen and Jong Cheol Jeong. 2007. Enhanced recursive feature elimination. In *Sixth International Conference on Machine Learning and Applications (ICMLA 2007)*, 429–435. DOI: 10.1109/ICMLA.2007.35.

[15] Graham Cluley. 2013. How twitter users can fake a verified account. https://nakedsecurity.sophos.com/2013/01/17/twitter-fake-verified-account/. Accessed: 2023-04-01. (2013).

[16] Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20, 3, 273–297.

[17] David R Cox. 1958. The regression analysis of binary sequences. *Journal of the Royal Statistical Society: Series B (Methodological)*, 20, 2, 215–232.

[18] Tobias Dam, Lukas Daniel Klausner, Damjan Buhov, and Sebastian Schrittwieser. 2019. Large-scale analysis of pop-up scam on typosquatting urls. In *Proceedings of the 14th International Conference on Availability, Reliability and Security* (ARES '19) Article 53. Association for Computing Machinery, Canterbury, CA, United Kingdom, 9 pages. ISBN: 9781450371643. DOI: 10.1145/3339252.3340332.

[19] Fred J. Damerau. 1964. A technique for computer detection and correction of spelling errors. *Commun. ACM*, 7, 3, (Mar. 1964), 171–176. DOI: 10.1145/363958.363994.

[20] George Danezis and Prateek Mittal. 2009. Sybilinfer: detecting sybil nodes using social networks. In *NDSS*.

[21] Ashish Dangwal. 2022. 'no weapon sales to israel': how a lockheed martin 'tweet' resulted in a loss of billions of dollars to us defense giant. https://eurasiantimes.com/no-weapons-sales-to-israel-how-a-lockheed-martin-tweet-resulted/. Accessed: 2022-12-01. (2022).

[22] Jesse Davis and Mark Goadrich. 2006. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd International Conference on Machine Learning* (ICML '06). Association for Computing Machinery, Pittsburgh, Pennsylvania, USA, 233–240. ISBN: 1595933832. DOI: 10.1145/1143844.1143874.

[23] Rocco De Nicola, Marinella Petrocchi, and Manuel Pratelli. 2021. On the efficacy of old features for the detection of new bots. *Information Processing & Management*, 58, (Nov. 2021), 102685. DOI: 10.1016/j.ipm.2021.102685.

[24] AnHai Doan, Alon Halevy, and Zachary Ives. 2012. 4 - string matching. In *Principles of Data Integration*. AnHai Doan, Alon Halevy, and Zachary Ives, (Eds.) Morgan Kaufmann, Boston, 95–119. ISBN: 978-0-12-416044-6. DOI: https://doi.org/10.1016/B978-0-12-416044-6.00004-1.

[25] Clare Duffy and Catherine Thorbecke. 2022. Elon musk said twitter has seen a 'massive drop in revenue' as more brands pause ads. https://www.edition.cnn.com/2022/11/04/tech/twitter-advertisers/index.html. Accessed: 2022-12-01. (2022).

[26] Ahmed ElAzab. 2016. Fake accounts detection in twitter based on minimum weighted feature. *World*.

[27] 2022. Elon musk's twitter lays off employees across the company. https://edition.cnn.com/2022/11/03/tech/twitter-layoffs/index.html. Accessed: 2022-12-01. (2022).

[28] External Data Source. 2018. Dnstwist. en. (2018). DOI: 10.23721/100/1504360.

[29] 2022. Facebook parent company meta will lay off 11,000 employees. https://edition.cnn.com/2022/11/09/tech/meta-facebook-layoffs/index.html. Accessed: 2022-12-01. (2022).

[30] [n. d.] Federal trade commission. https://www.ftc.gov/. Accessed: 2023-12-01. ().

[31] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, (Eds.) 2019. *Hyperparameter optimization. Automated Machine Learning: Methods, Systems, Challenges*. Springer International Publishing, Cham, 3–33. ISBN: 978-3-030-05318-5. DOI: 10.1007/978-3-030-05318-5_1.

[32] Qiang Fu, Bo Feng, Dong Guo, and Qiang Li. 2017. Combating the evolving spammers in online social networks. *Computers & Security*, 72, (Sept. 2017). DOI: 10.1016/j.cose.2017.08.014.

[33] Allison Gatlin. 2022. Eli lilly dives after fake twitter account promises free insulin; takes novo nordisk, sanofi with it. https://www.investors.com/news/technology/lly-stock-dives-taking-novo-sanofi-with-it-after-fake-twitter-account-promises-free-insulin/. Accessed: 2022-12-01. (2022).

[34] Priscila A. Gimenes, Norton T. Roman, and Ariadne M.B.R. Carvalho. 2015. Spelling Error Patterns in Brazilian Portuguese. *Computational Linguistics*, 41, 1, (Mar. 2015), 175–183. eprint: https://direct.mit.edu/coli/article-pdf/41/1/175/1805351/coli_a_00216.pdf. DOI: 10.1162/COLI_a_00216.

[35] Oana Goga, Giridhari Venkatadri, and Krishna P Gummadi. 2015. The doppelgänger bot attack: exploring identity impersonation in online social networks. In *Proceedings of the 2015 internet measurement conference*, 141–153.

[36] Chris Grier, Kurt Thomas, Vern Paxson, and Chao Michael Zhang. 2010. @spam: the underground on 140 characters or less. In *CCS '10*.

[37] Enzo Grossi and Massimo Buscema. 2008. Introduction to artificial neural networks. *European journal of gastroenterology & hepatology*, 19, (Jan. 2008), 1046–54. DOI: 10.1097/MEG.0b013e3282f198a0.

[38] Nuno Guimaraes, Alvaro Figueira, and Luis Torgo. 2020. Knowledge-based reliability metrics for social media accounts. In (Nov. 2020). DOI: 10.5220/0010140403390350.

[39] Drew Harwell. 2022. A fake tweet sparked panic at eli lilly and may have cost twitter millions. https://www.washingtonpost.com/technology/2022/11/14/twitter-fake-eli-lilly/. Accessed: 2022-12-01. (2022).

[40] Tin Kam Ho. 1995. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*. Vol. 1. IEEE, 278–282.

[41] Kris Holt. 2013. How to spot a fake verified twitter account. https://www.dailydot.com/unclick/how-to-spot-fake-verified-twitter/. Accessed: 2023-04-01. (2013).

[42] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. 2019. Squeeze-and-excitation networks. (2019). arXiv: 1709.01507 [cs.CV].

[43] Yangyu Hu, Haoyu Wang, Ren He, Li Li, Gareth Tyson, Ignacio Castro, Yao Guo, Lei Wu, and Guoai Xu. 2020. Mobile app squatting. In (Apr. 2020), 1727–1738. DOI: 10.1145/3366423.3380243.

[44] Pik-Mai Hui, Kai-Cheng Yang, Christopher Torres-Lugo, Zachary Monroe, Marc McCarty, Benjamin Serrette, Valentin Pentchev, and Filippo Menczer. 2019. Botslayer: real-time detection of bot amplification on twitter. *Journal of Open Source Software*, 4, (Oct. 2019), 1706. DOI: 10.21105/joss.01706.

[45] [n. d.] Information commissioner's office. https://ico.org.uk/. Accessed: 2023-12-01. ().

[46] [n. d.] Internet archive: wayback machine. https://archive.org/web/. Accessed: 2021-12-5. ().

[47] ItalianPostNews. 2022. Twitter, from apple to tesla the fake tweets with the "blue check" that have become memes. https://www.italianpost.news/twitter-from-apple-to-tesla-the-fake-tweets-with-the-blue-check-that-have-become-memes/. Accessed: 2022-12-01. (2022).

[48] Lei Jin, Daniel Takabi, and James Joshi. 2011. Towards active detection of identity clone attacks on online social networks. In (Feb. 2011), 27–38. DOI: 10.1145/1943513.1943520.

[49] 2017. Kaggle bots dataset. https://www.kaggle.com/vikasg/russian-troll-tweets. Accessed: 2021-10-14. (2017).

[50] [n. d.] Kaggle fake account dataset. https://www.kaggle.com/bitandatom/social-network-fake-account-dataset. Accessed: 2021-10-14. ().

[51] [n. d.] Kaggle popular accounts dataset. https://www.kaggle.com/parulpandey/100-mostfollowed-twitter-accounts-as-of-dec2019. Accessed: 2021-09-30. ().

[52] [n. d.] Kaggle spammer dataset. https://www.kaggle.com/free4ever1/instagram-fake-spammer-genuine-accounts. Accessed: 2021-10-14. ().

[53] SE Kelly, I Bourgeault, and R Dingwall. 2010. The sage handbook of qualitative methods in health research. *R. ingwall R. De Vries & I. Bourgeault (Eds.), London: Sage*.

[54] Panagiotis Kintis, Najmeh Miramirkhani, Charles Lever, Yizheng Chen, Rosa Romero-Gómez, Nikolaos Pitropakis, Nick Nikiforakis, and Manos Antonakakis. 2017. Hiding in plain sight: a longitudinal study of combosquatting abuse, (Aug. 2017). DOI: 10.1145/3133956.3134002.

[55] Georgios Kontaxis, Iasonas Polakis, Sotiris Ioannidis, and Evangelos Markatos. 2011. Detecting social network profile cloning. In (Apr. 2011), 295–300. DOI: 10.1109/PERCOMW.2011.5766886.

[56] Deepak Kumar, Riccardo Paccagnella, Paul Murley, Eric Hennenfent, Joshua Mason, Adam Bates, and Michael Bailey. 2018. Skill squatting attacks on amazon alexa. In *27th USENIX Security Symposium (USENIX Security 18)*. USENIX Association, Baltimore, MD, (Aug. 2018), 33–47. ISBN: 978-1-939133-04-5. https://www.usenix.org/conference/usenixsecurity18/presentation/kumar.

[57] APSS Lab. 2023. Squad. https://github.com/APSS-Imperial/SQUAD. (2023).

[58] APSS Lab. 2023. Squad framework. https://sites.google.com/view/squad-framework/home. (2023).

[59] Kyumin Lee, James Caverlee, and Steve Webb. 2010. Uncovering social spammers: social honeypots + machine learning. In (SIGIR '10). Association for Computing Machinery, Geneva, Switzerland, 435–442. ISBN: 9781450301534. DOI: 10.1145/1835449.1835522.

[60] Sangho Lee and Jong Kim. 2013. Warningbird: a near real-time detection system for suspicious urls in twitter stream. *Dependable and Secure Computing, IEEE Transactions on*, 10, (May 2013), 183–195. DOI: 10.1109/TDSC.2013.3.

[61] Let's Encrypt. 2017. Let's encrypt — free ssl/tls certificates. https://letsencrypt.org. Accessed: 2023-06-01. (2017).

[62] Steven Loria. 2018. Textblob documentation. *Release 0.15*, 2.

[63] Michal Majka. 2019. *naivebayes: High Performance Implementation of the Naive Bayes Algorithm in R*. R package version 0.9.7. https://CRAN.R-project.org/package=naivebayes.

[64] Enrico Mariconti, Jeremiah Onaolapo, Syed Ahmad, Nicolas Nikiforou, Manuel Egele, Nick Nikiforakis, and Gianluca Stringhini. 2016. Why allowing profile name reuse is a bad idea. In (Apr. 2016), 1–6. DOI: 10.1145/2905760.2905762.

[65] Enrico Mariconti, Jeremiah Onaolapo, Syed Sharique Ahmad, Nicolas Nikiforou, Manuel Egele, Nick Nikiforakis, and Gianluca Stringhini. 2017. What's in a name? understanding profile name reuse on twitter. In *Proceedings of the 26th International Conference on World Wide Web*, 1161–1170.

[66] Mary McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica : časopis Hrvatskoga društva medicinskih biokemičara / HDMB*, 22, (Oct. 2012), 276–82. DOI: 10.11613/BM.2012.031.

[67] 2009. *K-nearest neighbor classification. Data Mining in Agriculture*. Springer New York, New York, NY, 83–106. ISBN: 978-0-387-88615-2. DOI: 10.1007/978-0-387-88615-2_4.

[68] Nick Nikiforakis, Marco Balduzzi, Lieven Desmet, Frank Piessens, and Wouter Joosen. 2014. Soundsquatting: uncovering the use of homophones in domain squatting. In (Oct. 2014), 291–308. ISBN: 978-3-319-13256-3. DOI: 10.1007/978-3-319-13257-0_17.

[69] 2023. Oberlo. https://www.oberlo.com/blog/twitter-statistics. Accessed: 2023-11-01. (2023).

[70] Richard Oentaryo, Arinto Murdopo, Philips Kokoh Prasetyo, and Ee-Peng Lim. 2016. On profiling bots in social media. In (Sept. 2016). DOI: 10.1007/978-3-319-47880-7.

[71] Lawrence A Palinkas, Sarah M Horwitz, Carla A Green, Jennifer P Wisdom, Naihua Duan, and Kimberly Hoagwood. 2015. Purposeful sampling for qualitative data collection and analysis in mixed method implementation research. *Administration and policy in mental health and mental health services research*, 42, 533–544.

[72] F. Pedregosa et al. 2011. Scikit-learn: machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

[73] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, (Oct. 2014), 1532–1543. DOI: 10.3115/v1/D14-1162.

[74] Adrian Rauchfleisch and Jonas Kaiser. 2020. The false positive problem of automatic bot detection in social science research. *PLoS ONE*, 15, (Oct. 2020). DOI: 10.1371/journal.pone.0241045.

[75] Nicholas Reimann and Carlie Porterfield. 2022. Musk says apple cutting twitter ads—here are the other companies rethinking their ties. https://www.forbes.com/sites/nicholasreimann/2022/11/28/musk-says-apple-cutting-twitter-ads-here-are-the-other-companies-rethinking-their-ties/?sh=5efc41b77032. Accessed: 2022-12-01. (2022).

[76] Tech Report. [n. d.] The top 50 most popular followed x / twitter accounts. https://techreport.com/statistics/top-most-followed-x-twitter-accounts/. Accessed: 2023-12-01. ().

[77] Ellen Riloff, Siddharth Patwardhan, and Janyce Wiebe. 2006. Feature subsumption for opinion analysis. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, 440–448.

[78] Mohsen Sayyadiharikandeh, Onur Varol, Kai-Cheng Yang, Alessandro Flammini, and Filippo Menczer. 2020. Detection of novel social bots by ensembles of specialized classifiers. *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, (Oct. 2020). DOI: 10.1145/3340531.3412698.

[79] Sivanesh Seelan, K. Kavin, and A. Hassan. 2013. Frustrate twitter from automation: how far a user can be trusted? In (Aug. 2013), 1–5. ISBN: 978-1-4673-5703-6. DOI: 10.1109/ICHCI-IEEE.2013.6887787.

[80] Chengcheng Shao, Giovanni Ciampaglia, Onur Varol, Alessandro Flammini, and Filippo Menczer. 2017. The spread of fake news by social bots, (July 2017).

[81] Gaurav Sood. 2021. *virustotal: R Client for the virustotal API*. R package version 0.2.2.

[82] Gianluca Stringhini, Christopher Kruegel, and Giovanni Vigna. 2010. Detecting Spammers on Social Networks. In *Proceedings of the 26th Annual Computer Security Applications Conference* (ACSAC). ACM, New York, NY, USA, 1–9. ISBN: 978-1-4503-0133-6. DOI: 10.1145/1920261.1920263.

[83] Sysomos.com. [n. d.] Inside twitter: an in-depth look inside the twitter world. https://www.key4biz.it/files/000270/00027033.pdf. Accessed: 2023-06-01. ().

[84] Janos Szurdi and Nicolas Christin. 2017. Email typosquatting. In (Nov. 2017). DOI: 10.1145/3131365.3131399.

[85] Janos Szurdi, Balazs Kocso, Gabor Cseh, Jonathan Spring, Mark Felegyhazi, and Chris Kanich. 2014. The long "taile" of typosquatting domain names. In (Jan. 2014).

[86] 2020. *Characterizing social bots spreading financial disinformation*. (July 2020), 376–392. ISBN: 978-3-030-49569-5. DOI: 10.1007/978-3-030-49570-1_26.

[87] Timm Teubner and Sonia Camacho. 2023. Facing reciprocity: how photos and avatars promote interaction in micro-communities. *Group Decision and Negotiation*, 32, 2, 435–467.

[88] 2022. The digital services act package. https://digital-strategy.ec.europa.eu/en/policies/digital-services-act-package. Accessed: 2022-12-01. (2022).

[89] Kurt Thomas, Chris Grier, Justin Ma, Vern Paxson, and Dawn Song. 2011. Design and evaluation of a real-time url spam filtering service. In (May 2011), 447–462. DOI: 10.1109/SP.2011.25.

[90] Twitter. 2022. About twitter blue. https://help.twitter.com/en/using-twitter/twitter-blue. Accessed: 2022-12-01. (2022).

[91] Twitter. [n. d.] Academic research access deprecated. https://twitter.com/TwitterDev/status/1641222788911624192. Accessed: 2023-03-30. ().

[92] [n. d.] Twitter - bug boundy program. https://hackerone.com/twitter?type=team. Accessed: 2023-03-25. ().

[93] [n. d.] Twitter - country settings. https://help.twitter.com/en/managing-your-account/how-to-change-country-settings. Accessed: 2021-11-21. ().

[94] [n. d.] Twitter - country withheld content. https://help.twitter.com/en/rules-and-policies/tweet-withheld-by-country. Accessed: 2021-11-21. ().

[95] [n. d.] Twitter - rules. https://help.twitter.com/en/safety-and-security/report-twitter-impersonation. Accessed: 2023-04-03. ().

[96] [n. d.] Twitter - rules. https://help.twitter.com/en/rules-and-policies/twitter-rules.html. Accessed: 2023-04-03. ().

[97] [n. d.] Twitter - rules and policies. https://help.twitter.com/en/rules-and-policies/notices-on-twitter. Accessed: 2021-11-19. ().

[98] [n. d.] Twitter - suspension rules. https://blog.twitter.com/en_us/topics/company/2020/suspension. Accessed: 2021-11-19. ().

[99] [n. d.] Twitter academic api. https://developer.twitter.com/en/products/twitter-api/academic-research. Accessed: 2021-11-05. ().

[100] [n. d.] Twitter badge. https://help.twitter.com/en/managing-your-account/about-twitter-verified-accounts. Accessed: 2021-10-16. ().

[101] [n. d.] Twitter bots. https://blog.twitter.com/en_us/topics/company/2020/bot-or-not. Accessed: 2021-10-27. ().

[102] 2021. Twitter get-users. https://developer.twitter.com/en/docs/twitter-api/v1/accounts-and-users/follow-search-get-users/api-reference/get-users-search. Accessed: 2021-10-16. (2021).

[103] 2023. Twitter policies. https://help.twitter.com/en/rules-and-policies/twitter-impersonation-and-deceptive-identities-policy. Accessed: 2023-05-10. (2023).

[104] [n. d.] Twitter tweet-lookup. https://github.com/twitterdev/Twitter-API-v2-sample-code/tree/main/Tweet-Lookup. Accessed: 2021-12-01. ().

[105] 2016. Twitter user gender classification. https://www.kaggle.com/datasets/crowdflower/twitter-user-gender-classification. Accessed: 2023-06-01. (2016).

[106] [n. d.] Twitter user-lookup. https://github.com/twitterdev/Twitter-API-v2-sample-code/blob/main/User-Lookup. Accessed: 2021-10-16. ().

[107] [n. d.] Twitter username policy. https://help.twitter.com/en/managing-your-account/twitter-username-rules. Accessed: 2021-10-01. ().

[108] [n. d.] Urlcrazy. https://morningstarsecurity.com/research/urlcrazy. Accessed: 2021-10-01. ().
[109] Jordan Valinsky. 2023. Elon musk rebrands twitter as x. https://edition.cnn.com/2023/07/24/tech/twitter-rebrands-x-elon-musk-hnk-intl/index.html. Accessed: 2023-07-24. (2023).
[110] Alex Wang. 2010. Don't follow me - spam detection in twitter. In (Jan. 2010), 142–151. DOI: 10.7312/wang15140-003.
[111] Jess Weatherbed. 2022. Elon musk says twitter will begin manually authenticating blue, grey, and gold accounts as soon as next week. https://www.theverge.com/2022/11/25/23477550/twitter-manual-verification-blue-checkmark-gold-grey. Accessed: 2022-12-01. (2022).
[112] [n. d.] Wikipedia feature scaling. https://en.wikipedia.org/wiki/Feature_scaling. Accessed: 2021-12-27. ().
[113] [n. d.] Wikipedia levenshtein distance. https://en.wikipedia.org/wiki/Levenshtein_distance. Accessed: 2021-10-18. ().
[114] Matthew L Williams, Pete Burnap, and Luke Sloan. 2017. Towards an ethical framework for publishing twitter data in social research: taking into account users' views, online context and algorithmic estimation. *Sociology*, 51, 6, 1149–1168.
[115] Xindong Wu et al. 2008. Top 10 algorithms in data mining. *Knowledge and information systems*, 14, 1, 1–37.
[116] Teng Xu et al. 2021. Deep entity classification: abusive account detection for online social networks. In *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, (Aug. 2021), 4097–4114. ISBN: 978-1-939133-24-3. https://www.usenix.org/conference/usenixsecurity21/presentation/xu-teng.
[117] Chao Yang, Robert Harkreader, and Guofei Gu. 2011. Die free or live hard? empirical evaluation and new design for fighting evolving twitter spammers. In vol. 8. (Sept. 2011), 318–337. ISBN: 978-3-642-23643-3. DOI: 10.1109/TIFS.2013.2267732.
[118] Morteza Yousefi Kharaji, Fatemeh Salehi Rizi, and Mohammad Khayyambashi. 2014. A new approach for finding cloned profiles in online social networks. *ACEEE International Journal on Network Security*, (June 2014).
[119] Koosha Zarei, Reza Farahbakhsh, and Noel Crespi. 2019. Deep dive on politician impersonating accounts in social media. In (Apr. 2019). DOI: 10.1109/ISCC47284.2019.8969645.
[120] Koosha Zarei, Reza Farahbakhsh, Noel Crespi, and Gareth Tyson. 2020. Impersonation on social media: a deep neural approach to identify ingenuine content. In (Oct. 2020). DOI: 10.1109/ASONAM49781.2020.9381437.
[121] Chao Michael Zhang and Vern Paxson. 2011. Detecting and analyzing automated activity on twitter. In *PAM*.
[122] Nan Zhang, Xianghang Mi, Xuan Feng, XiaoFeng Wang, Yuan Tian, and Feng Qian. 2019. Dangerous skills: understanding and mitigating security risks of voice-controlled third-party functions on virtual personal assistant systems. In *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1381–1396.
[123] Xianghan Zheng, Zhipeng Zeng, Zheyi Chen, Yuanlong Yu, and Chunming Rong. 2015. Detecting spammers on social networks. *Neurocomputing*, 42, (Feb. 2015). DOI: 10.1016/j.neucom.2015.02.047.

A USERNAMECRAZY'S GENERATION MODELS

In Section 3.1 we introduced UsernameCrazy's taxonomy of generation models. Here we briefly describe each model.

Vowel Insertion: Inserts an extra and same vowel character when it finds one, e.g., '@AxlRose' into '@AaxlRose'.

Double Character Insertion: After finding two consecutive and identical characters, it inserts the same character next to them, e.g., '@CNNbrk' into '@CNNNbrk'.

Number Insertion: Adds a number (the same or a different one) both in the end and at the beginning of a username, '@Cristiano' into '@Cristiano21' or '@9Cristiano'.

Underscore Insertion: Inserts an underscore both in the end and at the beginning of a username, e.g., '@NBA' into '@NBA_' and '@_NBA'.

Vowel Deletion: Deletes a vowel character when it finds one, e.g., '@BarackObama' into '@BrackObama'.

Double Character Deletion: Deletes two consecutive and identical characters, e.g., '@Twitter' into '@Twier'.

Number Deletion: Deletes all the numbers of the username, one each time, starting both from the end and the beginning of a username, e.g., '@AndresIniesta8' into '@AndresIniesta'.

Underscore Deletion: Deletes an underscore when it finds one, e.g., '@Ricky_Martin' into '@RickyMartin'.

Vowel Character Substitution: Replaces a vowel with one of the rest vowel characters each time, e.g., '@BarackObama' into '@BerackObama'.

Common Misspellings/Homoglyphs: Specific characters are being replaced according to various recognized misspelling patterns [9], e.g., '@BarackObama' into '@BarakObama'. In this experiment, we used a small number of common mistakes to remain time-efficient.

In Section 6.1 we analyze the effectiveness of UsernameCrazy. Table 9 shows the number of the generated, active, suspended and active impersonator variants for each of the 10 most popular users.

B VGGFACE2 & THRESHOLDS

In Section 5.1 we described how we ensured the accuracy of VGGFace2 stays at a high enough score. Table 10 depicts the accuracy results of VGGFace2 with different thresholds. We observe that the highest test accuracy on our dataset was 85.93% when the threshold was set to 0.65.

C IMAGE SIMILARITY

In Section 3.2.4 we presented our image similarity experiment. Our image similarity algorithm is listed on Algorithm 1.

Algorithm 1 Image Similarity Algorithm

Input All usernames (original + generated)
Output Similar images

procedure IMAGE_SIMILARITY
  users[] ← *original usernames / initial seed*
  if *user not a celebrity* **then**
    Remove user from users
  for user in range(len(users)) **do**
    orig_img[user] ← *download image of celebrity*
  for user in range(len(orig_img)) **do**
    if *image[user] does not have a face* **then**
      Remove orig_img[user] from orig_img[]
      Remove user from users
  gen_users[][] ← *generated username variants (UVs)*
                ▷ 1st array: all the remaining users
                ▷ 2nd: all the generated UVs of a user
  for i in range(len(gen_users)) **do**
    for j in range(len(gen_users[i])) **do**
      gen_imgs[i][j] ← *download image of user*

                ▷ there are multiple images in each row
  for i in range(len(gen_imgs)) **do**
    for j in range(len(gen_imgs[i])) **do**
                ▷ calculation of distance between embeddings
      if distance($orig\_img[i]$, $gen\_imgs[i][j]$) < threshold **then**
        return Pair of similar images
      else
        No similarity

D PROFILE NAME SIMILARITY

In Section 3.2.4 we described our profile name similarity experiment. Our profile name similarity algorithm is described in Algorithm 2.

Table 9: Competency of *URLCrazy*, *AppCrazy* and *UsernameCrazy*.

Usernames	URLCrazy				AppCrazy				UsernameCrazy			
	# of Generated Usernames	# of Active Squatted Usernames	# of Suspended Squatted Usernames	# of Active Impersonators	# of Generated Usernames	# of Active Squatted Usernames	# of Suspended Squatted Usernames	# of Active Impersonators	# of Generated Usernames	# of Active Squatted Usernames	# of Suspended Squatted Usernames	# of Active Impersonators
@barackobama	135	20	6	0	57	35	12	0	9,800	307	104	1
@katyperry	118	17	9	0	25	15	5	0	5,992	640	500	7
@justinbieber	170	34	4	3	60	52	7	5	9,010	555	110	20
@rihanna	94	21	3	0	38	32	2	1	13,086	1,113	199	9
@taylorswift13	172	22	9	2	36	25	8	2	5,766	102	23	5
@cristiano	131	34	1	0	49	46	1	0	17,687	1,714	134	1
@ladygaga	96	18	2	0	32	26	6	0	6,821	954	287	10
@theellenshow	168	5	6	1	51	13	8	2	6,999	46	62	7
@youtube	101	30	2	3	41	31	9	5	13,057	1,078	353	12
@jtimberlake	156	15	8	1	47	18	15	2	8,750	120	129	16
Total (10 users)	**1,274**	**215 (16.87%)**	**49 (3.84%)**	**10**	**436**	**293 (67.20%)**	**73 (16.74%)**	**17**	**96,968**	**6,629 (6.83%)**	**1,901 (1.96%)**	**88**

Table 10: Accuracy of VGGFace2 with various thresholds. Setting the threshold value to 0.65, the model has the best accuracy.

VGGFace2				
Threshold	**Accuracy**	**# of Images**	**# of FP**	**# of FN**
0.5	82.81%	128	9	13
0.6	84.37%	128	8	12
0.65	85.93%	128	6	12
0.7	85.15%	128	7	12

Algorithm 2 Profile Name Similarity Algorithm

Input All usernames (original + generated)
Output Similar profile names

procedure PROFILE_NAME_SIMILARITY
  users[] ← *original usernames / initial seed*
  gen_users[][] ← *generated UVs* ▷ 1st array: all the users
  ▷ 2nd: all the generated UVs of a user
  for i in range(len(gen_users)) **do**
    try
    ▷ For space limitations we assume that a profile name can be splitted by ' '
      prof_name ← *gen_users[i][0].split(' ')*
      splitted_name ← *gen_users[i][0].split(' ')*
      for j in range(1, len(gen_users[i])) **do**
        if *prof_name* == *gen_users[i][j]* **then**
          return Exact match
        else if *gen_users[i][0]* in *gen_users[i][j]* AND
          gen_users[i][j] not in *prof_name* **then**
          return Exact match plus at least one character extra
    ▷ For space limitations we assume that a profile name consists of 2 words
        else if *splitted_name[0]* in *gen_users[i][j]* AND
          splitted_name[1] not in *gen_users[i][j]* **then**
          return At least one word of the *prof_name* is appeared but in a different order
        else
          return Other
    catch ▷ We treat the profile name as one word only
      for j in range(1, len(gen_users[i])) **do**
        if *prof_name* == *gen_users[i][j]* **then**
          return Exact match
        else if *gen_users[i][0]* in *gen_users[i][j]* AND
          gen_users[i][j] not in prof_name **then**
          return Exact match plus at least one character extra
    end try

E NON-POPULAR ACCOUNTS

In Section 7 we applied SQUAD to a set of non-popular users. Here, Table 11 presents the classification results of SQUAD on the 15 non-popular accounts. We observe that all the 61 active accounts with a face in their profile photo were classified as *benign* by SQUAD. Lasty, SQUAD returned only 1 *false negative* case.

Table 11: SQUAD results on non-popular accounts.

SQUAD - Non Popular Accounts											
Original Account	**Generated Accounts**	**Accounts with ED 1-3**	**Active Accounts**	**Active Accounts with ED 1-3**	**Suspended Accounts**	**Suspended Accounts with ED 1-3**	**Followees**	**Followers**	**Classified Malicious by SQUAD**	**Classified Benign by SQUAD**	**Active Accounts w/o Face in Profile Image**
Account1	11,876	6,045 (50.9%)	3 (0.02%)	0 (0%)	0 (0%)	0 (0%)	502	186	0 (0%)	1 (33.3%)	2 (66.6%)
Account2	5,975	3,225 (53.9%)	1 (0.01%)	1 (100%)	0 (0%)	0 (0%)	195	15	0 (0%)	0 (0%)	1 (100%)
Account3	12,929	4,718 (36.4%)	44 (0.34%)	30 (68.1%)	2 (0.01%)	2 (100%)	33	94	0 (0%)	19 (43.2%)	25 (56.8%)
Account4	6,983	3,179 (45.5%)	67 (0.96%)	40 (59.7%)	2 (0.03%)	2 (100%)	333	69	0 (0%)	23 (34.3%)	44 (65.7%)
Account5	11,612	5,749 (49.5%)	1 (0.01%)	1 (100%)	0 (0%)	0 (0%)	0	0	0 (0%)	1 (100%)	0 (0%)
Account6	1,002	916 (91.4%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	128	145	0 (0%)	0 (0%)	0 (0%)
Account7	4,541	1,951 (42.9%)	7 (0.15%)	7 (100%)	0 (0%)	0 (0%)	4	4	0 (0%)	3 (42.9%)	4 (57.1%)
Account8	1,454	1,287 (88.5%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	173	65	0 (0%)	0 (0%)	0 (0%)
Account9	13,756	6,883 (50%)	23 (0.16%)	18 (78.3%)	1 (0.01%)	1 (100%)	0	0	0 (0%) $\#FP = 0$	6 (26.1%) $\#FN = 1$	17 (73.9%)
Account10	3,116	2,860 (91.8%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	121	129	0 (0%)	0 (0%)	0 (0%)
Account11	4,159	2,854 (68.6%)	3 (0.07)	3 (100%)	0 (0%)	0 (0%)	79	140	0 (0%)	1 (33.3%)	2 (66.6%)
Account12	1,779	1,644 (92.4%)	2 (0.11%)	2 (100%)	0 (0%)	0 (0%)	14	6	0 (0%)	2 (100%)	0 (0%)
Account13	8,164	3,885 (47.6%)	7 (0.09%)	5 (71.4%)	0 (0%)	0 (0%)	490	119	0 (0%)	4 (57.1%)	3 (42.9%)
Account14	6,360	4,627 (72.8%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	78	28	0 (0%)	0 (0%)	0 (0%)
Account15	7,883	4,524 (57.4%)	1 (0.01%)	0 (0%)	0 (0%)	0 (0%)	62	38	0 (0%)	1 (100%)	0 (0%)
Total	101,589	54,347 (53.5%)	159 (15.7%)	107 (67.3%)	5 (4.92%)	5 (100%)	2,212	1,038	0 (0%) $\#FP = 0$	61 (38.4%) $\#FN = 1$	98 (61.6%)

F USERNAME CHARACTERISTICS

In Section 4.3 we analyzed the characteristics of our squatted usernames. Figure 8 further depicts the number of suspended and active accounts over different edit distance values. We observe that the highest number of both active and suspended profiles are squatted usernames with edit distance one to three characters from their target.

G SQUAD'S PERFORMANCE

In Section 6.2 we discussed the training and evaluation metrics applied for SQUAD as well as the overall performance of all the classifiers. Here we further present the actual results of the classification performance of all the algorithms. Table 12 depicts the classification performance results of the 7 models. Random Forest exhibits the best performance in identifying suspicious accounts.

Figure 9 shows the '*Mean AUC*' score of all classification models.

H ETHICS: RESPONSIBLE DISCLOSURE

We believe it should be the responsibility of the platform to integrate automated processes to identify impersonators and suspicious accounts. We reported all the manually verified *impersonators* (see

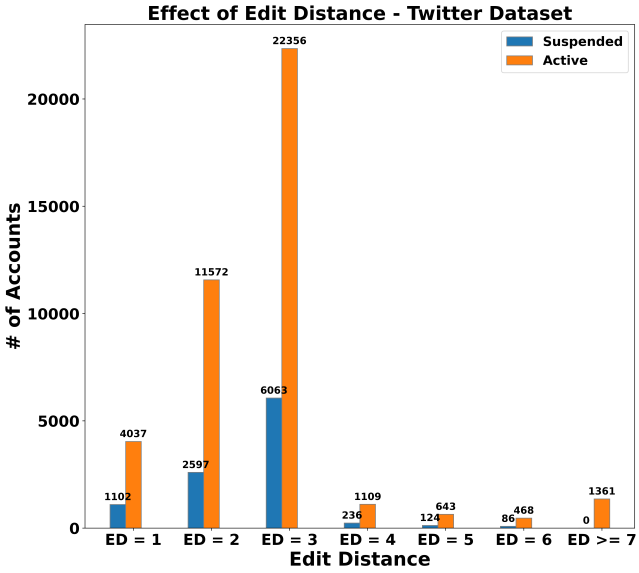

Figure 8: The number of active and suspended accounts across different edit distance categories.

Table 12: Classification performance. (P) shows the positive class.

Models	Precision	Recall	F1 - Score
Random Forest			
(P) Suspicious	0.97	0.94	0.95
(P) Benign	0.91	0.95	0.93
Naive Bayes			
(P) Suspicious	0.98	0.82	0.89
(P) Benign	0.78	0.97	0.86
Logistic Regression			
(P) Suspicious	0.94	0.90	0.92
(P) Benign	0.85	0.91	0.88
SVM			
(P) Suspicious	0.95	0.90	0.92
(P) Benign	0.85	0.93	0.89
Decision Trees			
(P) Suspicious	0.94	0.93	0.94
(P) Benign	0.90	0.91	0.90
K-Nearest Neighbor			
(P) Suspicious	0.93	0.86	0.89
(P) Benign	0.81	0.90	0.85
Neural Network			
(P) Suspicious	0.96	0.90	0.93
(P) Benign	0.86	0.94	0.90

Section 4.4) to X via a formal report at HackerOne program [92]. X acknowledged the problem and marked our report as *Informative*. Unfortunately, X's response (see below) suggests that the platform simply relies on weak crowdsourcing methods [95] for identifying impersonation attempts even though it is aware of the problem. Such methods are known to suffer from erroneous reports and manual verification of crowdsourced reports does not scale.

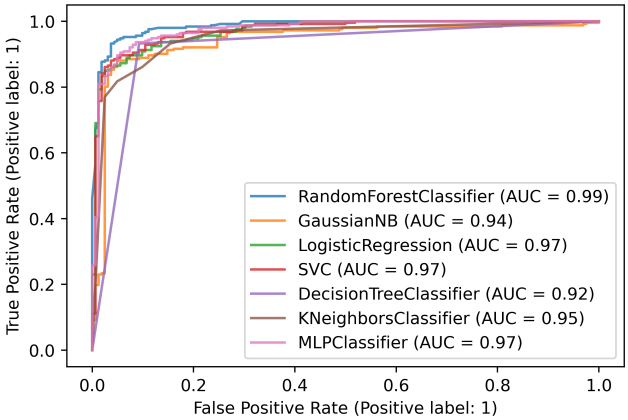

Figure 9: ROC Curve of all the classification models.

We share below X's official response (verbatim) to our report which included all manually verified and active impersonation accounts: "Thank you for your report. Please bear in mind that our HackerOne program [92] is for the reporting of explicit security vulnerabilities in Twitter services. We are already aware that accounts impersonating high-profile Twitter users is an issue on our platform. For this reason, we already provide additional controls [95] which allow users to report these accounts which may be in violation of Twitter rules [96]. While this is an issue we are already aware of, this report does not appear to concern an exploitable vulnerability in Twitter services. Furthermore, as we already have controls in place to allow Twitter users to report impersonator accounts, we will be closing this report as 'Informative'. Regardless, we do appreciate your interest in our program, and we encourage you to continue hunting for security vulnerabilities. Thank you for thinking of Twitter security."