UNIVERSITÄT KOBLENZ · LANDAU
Fachbereich 4: Informatik

WeST People and Knowledge Networks
Institute for Web Science and Technologies

Enriching the feature space of transfer learning in user analysis in online social networks

Bachelorarbeit

zur Erlangung des Grades einer Bachelor of Science (B.Sc.)
im Studiengang Informatik

vorgelegt von
Benjamin Zill

Erstgutachter:	Prof. Dr. Steffen Staab Institute for Web Science and Technologies
Zweitgutachter:	Jun Sun Institute for Web Science and Technologies

Koblenz, im November 2018

Erklärung

Hiermit bestätige ich, dass die vorliegende Arbeit von mir selbstständig verfasst wurde und ich keine anderen als die angegebenen Hilfsmittel – insbesondere keine im Quellenverzeichnis nicht benannten Internet-Quellen – benutzt habe und die Arbeit von mir vorher nicht in einem anderen Prüfungsverfahren eingereicht wurde. Die eingereichte schriftliche Fassung entspricht der auf dem elektronischen Speichermedium (CD-Rom).

	Ja	Nein
Mit der Einstellung dieser Arbeit in die Bibliothek bin ich einverstanden.	☒	☐
Der Veröffentlichung dieser Arbeit im Internet stimme ich zu.	☒	☐
Der Text dieser Arbeit ist unter einer Creative Commons Lizenz (CC BY-SA 4.0) verfügbar.	☒	☐
Der Quellcode ist unter einer GNU General Public License (GPLv3) verfügbar.	☒	☐
Die erhobenen Daten sind unter einer Creative Commons Lizenz (CC BY-SA 4.0) verfügbar.	☒	☐

30.11.2018, Bad Hönningen

(Ort, Datum) (Unterschrift)

Zusammenfassung

Wikipedia ist die größte online Enzyklopädie, welche für jeden frei zugänglich ist und erweitert werden kann. Für die Nutzer, die innerhalb dieser Enzyklopädie Beiträge erstellen, existiert ein soziales Netzwerk. Nutzer sind unterteilt in sogenannte Rollen, welche aus normalen Nutzern, Administratoren und funktionalen Bots bestehen. In dem Netzwerk können Nutzer Kritik, Anregungen oder einfache Beiträge auf die Beitragsseiten anderer Nutzer verfassen. Jede Sprachausgabe besitzt jeweils ein eigenes Netzwerk. In dieser Bachelorarbeit analysieren wir Merkmale der drei verschiedenen Rollen, um diese aus einem Wikipedia Netzwerk einer bestimmten Sprache zu lernen und in einem anderen Wikipedia Netzwerk anzuwenden, um Bots zu identifizieren. Dabei werden Zeitstempel der erstellten Beiträge analysiert um Auffälligkeiten in Bezug auf Abstand zwei aufeinander folgenden Nachrichten, die Anzahl der Nachrichten innerhalb eines Zeitfensters und reguläres verhalten festzustellen. Dabei ist festzustellen das innerhalb von Netzwerken in denen funktionale Bots verwendet werden eine Auffälligkeit dieser Merkmale vorhanden ist.

Abstract

Wikipedia is the biggest, free online encyclopaedia that can be expanded by anyone. For the users, who create content on a specific Wikipedia language edition, a social network exists. In this social network users are categorised into different roles. These are normal users, administrators and functional bots. Within the networks, a user can post reviews, suggestions or send simple messages to the "talk page" of another user. Each language in the Wikipedia domain has this type of social network. In this thesis characteristics of the three different roles are analysed in order to learn how they function in one language network of Wikipedia and apply them to another Wikipedia network to identify bots. Timestamps from created posts are analysed to reveal noticeable characteristics referring to continuous messages, message rates and irregular behaviour of a user are discovered. Through this process we show that there exist differences between the roles for the mentioned characteristics.

Contents

1 Introduction

The topic of analysing and identifying users in social networks has received increasing attention in recent years. Based on different incidences in the last decade, e.g. advertisements, spam and fake news used for propaganda with political purposes in the 2010 US midterm elections [28][26], the need to identify bots within various social networks has become clear. When seeking to differentiate and classify a user as a bot or a human being, it is important to note that human users in social networks exhibit special behaviours. For instance, human users send a manageable number of messages, connect to other people, consume or spread information and create content. Gathering user-specific data about these behaviours allows an analysis to be made between this user and other users of a social network. Recent analysis of user profiles [1], the node structure of social graphs [2][3] or classification based on messages or post content [1] are common methods for classifying or labelling different users based on their available information using machine learning [4]. In this thesis we identify bot users within the *Wiki-Talk* datasets using a collection of user messages from the Wikipedia social network. In doing this we extend the work of Jun and colleagues [5] by developing a function to read and process message timestamps from users. Jun et al. used a structure based approach to classify administrator users within this social network. When using machine learning, problem arises when a dataset is too small to train a reliable classifier. To overcome this, Jun et al. used transfer learning with a feature transformation approach. We extend their classification by enriching the feature set (which consists of only structural features) with new features, which are derived by analysing the user behaviour. Our observation focus is on time stamp evaluation of messages to increase performance in classifying bots. Because computers are able to respond to events or activities more quickly than human beings, e.g. bots used in contract for difference trading to order contracts after analysing events, the approach assumes that bots have a higher message rate within smaller time frames, and a smaller time interval between their messages compared to human users. In addition, we investigate whether there are regular behaviours in messaging on specific days or at specific times. Based on these considerations, we develop three research questions related to the Wiki-Talk social network datasets that are answered in this thesis. 1) Are there significant low message intervals from bots? 2) Is there a noticeable number of messages within a small time frame from bots? 3) Can we observe regular messaging behaviour of bots?

After answering these questions by observing user behaviour we enrich the baseline feature set created by Jun et al. with new features and perform their transformation to a common feature space between different datasets, to support learning from one dataset to predict bot users within another.

This thesis is organized as follows. In Chapter 2 we introduce some background knowledge, about Wikipedia, the Wiki-Talk datasets, different metrics used and some basic background knowledge about graphs and structural properties of a node. Machine learning and transfer learning are also introduced, the ROC-AUC score for

measuring performance of a classifier and the method for preparing a dataset for testing our new features are also explained.
Chapter 3 presents the baseline study of Jun et al. as well as previous research on identifying bots using timestamp features.
In Chapter 4 we introduce our method for reading and processing the message timestamps of users and different techniques for presenting the social network graphs as a matrix. For the transfer learning scenario, a feature normalization and a transformation method is outlined.
Chapter 5 constitutes the main portion of this thesis. The implementation and results from the work of Jun et al. are presented for bot classification, which provides us with a lower bound of scores upon which we want to improve. We observe the datasets, investigate the intervals between messages and message rates and search for regular or irregular behaviour of sending messages within a specific time or day of the week for the different user roles.
In section 6, we first look how the created features perform within each language dataset and we select the best features for the final transfer learning by their ROC-AUC scores. Also, the final results are shown, and we discuss the selected features and achieved score.
Section 7 we present some thoughts for future work that can be done to improve this research.

2 Background

In this chapter we introduce the Wikipedia domain and the datasets that are used to observe behaviour and classify users in the experiment. In the user observation and feature creation presented later in this thesis we use two mean metrics and Shannon entropy, presented in Section 2.3. Basic definitions of graphs and node properties are given in Section 2.4. These are the structural features used in the experiment. Degree, indegree, outdegree, PageRank and local clustering coefficient are features from the baseline set developed by Jun et al. and k-core is the first enriched feature offered by this thesis. The final section of this chapter addresses the topic of machine learning and transfer learning. To measure the performance of the predictions we explain the ROC-AUC score as the evaluation metric. Finally, we explain the k-fold cross validation method used for splitting a dataset into various training and test parts.

2.1 Wikipedia

Wikipedia, a free online encyclopaedia with over 40 million [1] articles in 301 available languages, is the largest online encyclopaedia. In order to generate such a large number of articles, the contributions of many users are necessary. Over the last two decades, an enormous community has developed that includes normal users, who write articles for Wikipedia, and administrators, who verify the accuracy of articles if necessary. Every user has a so-called "talk page", where communication between different users takes place. Users are also separated into access levels [6] which this thesis uses to distinguish between three different roles. An administrator is an editor with high-level access. He or she can use specific tools in relation to articles, with functionalities such as delete, block, protect or revert. In Wikipedia there is also the role of bureaucrat. Individuals with this access level perform different actions on user accounts. In this thesis, we combine both levels into one classification of "administrator role". Bots are defined as automatic or semi-automatic users who perform actions on the content of articles. There are different types of bots and each has its own function. For example, the citation bot [7] adds digital object identifiers, ISBNs and other identifiers to article references. The third role is a normal user. Users with this role can, for example, create articles or participate in conversations on a user's talk page.

2.2 Datasets

The dataset used in this experiment is the Wiki-Talk dataset. This dataset contains all messages from a user's talk page. A total of 28 different sub-datasets are available and every set contains the Wiki-Talk community for a given language. Each sub-dataset consists of two different files. The Wiki-Talk group data file is comprised of two primary pieces of information: a given node ID, which corresponds to a given user, and his or her given role ID. As previously mentioned, there are three types of roles, i.e. normal user, bot and administrator.
The second file contains the Wiki-Talk messages. This data includes information about the timestamp of messages sent between users' talk pages. Timestamps are given in the format "yyyy-mm-ddThh:mm:ssZ", where "T" separates date from exact time and "Z" stands for time zone UTC. An overview for the datasets is given in Table 1. Every language is represented using an acronym according to ISO 639 code. These datasets are from the results of the work of J. Sun and J. Kunegis[8].

[1] As of August 2018 (https://en.wikipedia.org/wiki/Wikipedia)

Lang	# Nodes	# Edges	# Users	# Bots	# Admins	% User	% Bots	% Admins
ar	1,095,799	1,913,103	1,095,732	30	37	99.9939 %	0.0027%	0.0034%
bn	83,803	122,078	83,769	18	16	99.9594 %	0.0215%	0.0191 %
br	1,181	13,754	1,130	43	8	95.6816 %	3.641%	0.6774 %
ca	79,736	351,610	79,532	179	25	99.7442 %	0.2245%	0.0314%
cy	2,233	10,740	2,178	39	16	97.5369 %	1.7465%	0.7165%
de	519,403	6,729,794	518,829	328	246	99.8895 %	0.0631%	0.0474 %
el	40,254	190,279	40,176	58	20	99.8062 %	0.1441%	0.0497%
en	2,987,535	2,4981,163	2,985,944	278	1313	99.9467 %	0.0093%	0.0439%
eo	7,586	47,070	7435	130	21	98.0095 %	1.7137%	0.2768%
es	497,446	2,702,879	497,337	34	75	99.9781 %	0.0068%	0.0151%
eu	40,993	58,120	40,902	81	10	99.778 %	0.1976%	0.0244%
fr	1,420,367	4,641,928	1,420,107	97	163	99.9817 %	0.0068%	0.0115%
gl	8,097	63,809	8,071	12	14	99.6789 %	0.1482%	0.1729%
ht	536	1,530	504	32	0	94.0299 %	5.9701%	0.0%
it	863,846	3,067,680	863605	137	104	99.9721 %	0.0159%	0.012%
ja	397,635	1,031,378	397,535	51	49	99.9749 %	0.0128%	0.0123%
lv	41,424	73,900	41,356	57	11	99.8358 %	0.1376%	0.0266%
nds	23,132	27,432	23,071	56	5	99.7363 %	0.2421%	0.0216 %
nl	225,749	1,554,699	225,462	237	50	99.8729 %	0.105%	0.0221%
oc	3,144	11,059	3,089	51	4	98.2506 %	1.6221%	0.1272%
pl	155,820	1,358,426	155,650	55	115	99.8909 %	0.0353%	0.0738%
pt	541,355	2,424,962	541,086	205	64	99.9503 %	0.0379%	0.0118%
ru	457,017	2,282,055	456,850	77	90	99.9635 %	0.0168%	0.0197 %
sk	41,452	131,884	41,339	105	8	99.7274 %	0.2533%	0.0193%
sr	103,068	312,837	102,914	132	22	99.8506 %	0.1281%	0.0213%
sv	120,833	598,066	120,720	41	72	99.9065 %	0.0339%	0.0596 %
vi	338,714	607,087	338,568	123	23	99.9569 %	0.0363%	0.0068 8%
zh	1,219,241	2,284,546	1,219,071	93	77	99.9861 %	0.0076%	0.0063%

Table 1: Wiki-Talk datasets and 28 subsets with information on nodes, edges and role distribution.

2.3 Different metrics

2.3.1 Shannon entropy

Shannon entropy is a metric to measure uncertainty in a set of quantities. The entropy of a set of random variables X is defined by

$$H(X) = -\sum_{x \in X} p(x) log(p(x)), \tag{1}$$

where $0 \leq p(x) \leq 1$ and $p(x)$ is the probability of $x \in X$. This metric is used for measuring regularities or irregularities in users message behaviour. Additionally, we define $0 \cdot log(0) := 0$ for participants who did not send any message.

2.3.2 Mean

Arithmetic mean is used to calculate average value over a set of data. Given a set of values $x_i \in X$, with $x_i \in \mathbb{R}$, the arithmetic mean is calculated by

$$M(X) = \frac{x_1 + x_2 + .. + x_n}{n} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{2}$$

where $n \in \mathbb{N}$ is the total number of values in X.

Geometric mean is calculated differently from the arithmetic and finds central tendency by using the n-th root of the product from all values $x_i \in X$. It is calculated by

$$GM(X) = \sqrt[n]{x_1 \cdot x_2 \cdot .. \cdot x_n} \tag{3}$$

Again, $n \in \mathbb{N}$ is the total number of all elements within the set X. Considering that we work with the product of values, the geometrical mean is applicable only for x_i values that are not negative and are greater than zero.

The reason we use and test different means in this work is because of the presence of outliers in the given sets of values. Outliers have a significant impact on the central tendency value when calculated using arithmetic mean. In contrast, geometric mean is not influenced by single outliers. In certain cases, we want to include the influence of these outliers.

2.4 Introduction into graphs

In order to analyse a social network, we need to model it as a graph. We provide some basic definitions of a graph and its representation as an adjacency matrix in this section. In addition, some node properties are described that are later used as features for the machine learning model.

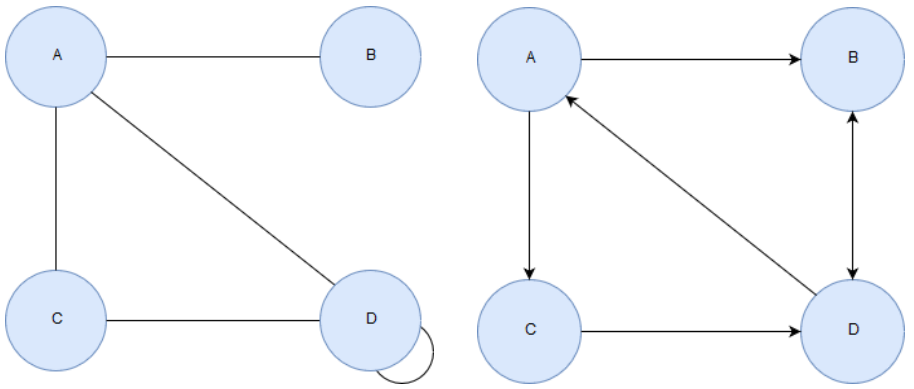
2.4.1 Definition of a graph

In graph theory there are various definitions of a graph. In this work we use a common and basic definition:

Definition 1. $G = (V, E)$

The graph G is a pair where V is the set of vertices or nodes and E is the multiset of edges or connection points. One edge refers to two nodes or to a single node itself, called a loop. Undirected graphs consist of edges with no direction, while directed graphs are those where the edge points to a node in one direction, also called orientation.

Figure 1a shows a simple undirected graph, with node labels A, B, C and D. The nodes are connected to each other by edges. Our node set for this graph is $V = \{A, B, C, D\}$. The edge set is $E = \{(A, B), (A, C), (A, D), (C, D), (D, D)\}$.



(a) A simple undirected graph with four nodes and one loop.

(b) A simple directed graph with four nodes.

Figure 1: Example of undirected and directed graph with four nodes.

Another common method for representing graphs is an adjacency matrix. An adjacency matrix for the undirected graph in 1a is presented in 2a, and an adjacency matrix for the directed graph in Figure 1b is given in Figure 2b.

The adjacency matrix is a square matrix with $|V|*|V|$, with n rows and m columns. A field marked with 1 means that the node of a row n is connected to the node of column m. A field marked with 0 means that there is no connection between these nodes.

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} \end{array}$$

(a) Adjacency matrix from Figure 1a.

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

(b) Adjacency matrix from Figure 1b.

Figure 2: Adjacency matrix.

Also we give a definition of subgraphs adapted by Bollobás [9].

Definition 2. *$G' = (V', E')$ is a subgraph of $G = (V, E)$ if $V' \subset V$ and $E' \subset E$. If this is the case, we write $G' \subset G$.*

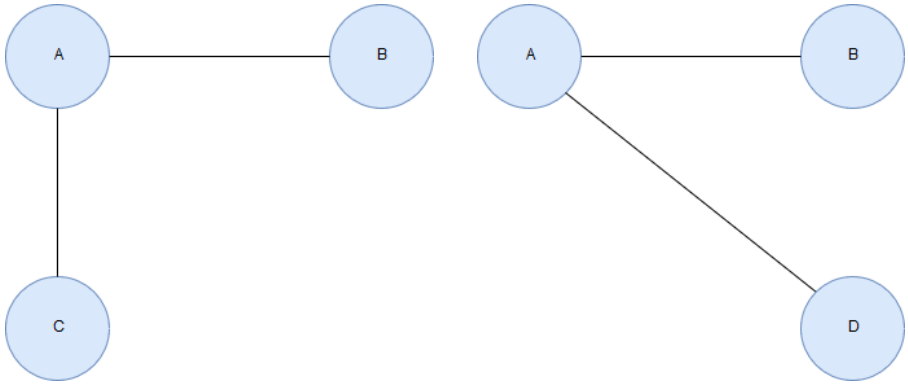


Figure 3: Two subgraphs from the graph in Figure 1a

Figure 3 shows two subgraphs from the graphs in Figure 1a. It can be seen that the subgraphs are parts of graphs, as they exhibit the same node edge combinations. If we also examine the adjacency matrix from Figure 1a and compare it with Figure 3a, we see in Figure 4 that the values of the nodes match.

With this knowledge we can represent a social network as a graph. Users are nodes and a message from user A to user B is represented as a directed edge. In cases where there is more than one message sent between the same users, the specific row and column field contain the number of edges from one node to another.

2.4.2 Node properties

Degree of a node is the number of edges connected to the node. In the undirected graph we have no distinction in degree. With directed graphs we have the outdegree of a node, which is the number of edges going from one node to other nodes.

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} \end{array}$$

(a) Adjacency matrix from Figure 1a.

$$\begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \end{array}$$

(b) Adjacency matrix from Figure 3a.

Figure 4: Adjacency matrix from Figure 1a and Figure 3a.

The indegree shows how many neighbour nodes have a connection to the specific node. In the experiment we define the feature indegree as the number of messages a user receives and outdegree as the number of messages a user sends. The degree represents all in- and outgoing messages of a user.

Cluster coefficient is a measurement of the transitivity in a network. There are different types of cluster coefficients, such as local, global or average cluster coefficients. For our experiment the local cluster coefficient is of importance, because we are interested in the node's information about the connectivity of its neighbourhood. The local cluster coefficient of a node v from the directed graph G describes the possibility of that node's neighbours being connected and thus creating a subgraph. The coefficient is measured as

$$c_i = \frac{|\{e_{jk} : v_j, v_k \in N_i, e_{jk} \in E|\}}{k_i(k_i - 1)} \tag{4}$$

where e_{jk} is the edge between node v_j and v_k from N_i. N_i is defined as

$$N_i = \{v_j : e_{ij} \in Eve_{ji} \in E\}, \tag{5}$$

the set of neighbours from node v_i. The total number of v_i edges is defined as k_i. The clustering coefficient is the difference in edges from the set of neighbours from node v. Within the social graph of Wikipedia, the cluster coefficient provides information about the communication between the neighbours of a user. A high cluster coefficient implies active communication around a user.

PageRank is an algorithm generally used to measure the importance of websites. By counting the number of links to a web page we can assume how important the given page is. Pages with a high PageRank score have many incoming or outgoing links from other pages with a high score. Pages with a small number of incoming links and thus with low importance have a low PageRank. The PageRank of a website i is calculated by the recursive function:

$$PR_i = \frac{1-d}{n} + d \sum_{j \in \{1,..,n\}} \frac{PR_j}{c_j}, \qquad 0 \leq d \leq 1 \tag{6}$$

PR_I for the website i is calculated by the sum of i neighbours PR_j. c_j is the number of links PR_j is referring to. The weight of PR_j is separated on these pages. To not let the weight left out to pages, that are not referring to other pages a damping d factor is needed. This d will be subtracted from the given page as a small amount of the weight $(1 - d)$ and equally distributed to all given pages. Since a collection of linked web pages can be also modelled as a network, the PageRank can also be computed for users in our social network and its calculated with the recursive function:

$$PR_{u_i} = \frac{1-d}{n} + d \sum_{j \in \{1,..,n\}} \frac{PR_{u_j}}{c_{u_j}}, \qquad 0 \leq d \leq 1 \tag{7}$$

PR_I for user u_i is calculated by the sum of all its neighbours u_j PageRank in proportion to u_j linked users. d is the damping factor, usually 0.85 [10].

K-Core number of a graph, also known as the degeneracy of a graph, is a value that provides information about central node groups. In social networks these nodes are important for spreading information within a network and building communities [11].

The degeneracy of a graph is the largest subgraph in which every vertex has a degree of at least k within the subgraph [12]. For social networks, this would be the largest group of people who are connected to each other with almost the same degree. Furthermore, every node has a specific k-core number of the subgraph in which it exists.

We adapt a formal definition for the k-core of a graph from O'Brien et al. [13]:

Definition 3. *The k-core of G, denoted by C_k, is the maximal induced subgraph of G with a minimum degree of at least k.*

Additionally, we take definitions for core numbers and the degeneracy of a graph from [13]:

Definition 4. *The core number of v, denoted by $k(v)$, is the largest $k >= 0$ such that $v \in C_k$.*

Definition 5. *The degeneracy of G is the largest k for which $|C_k| > 0$; a graph with degeneracy of at least D is said to be D-degenerate.*

Using Algorithm 1, we can evaluate core numbers from a given adjacency matrix of a graph. The algorithm is executed until every value of the matrix is zero. Beginning with $i = 0$, the algorithm searches for the node with degree at most i. The degree is the sum of row j. When a node's degree is lower than or equal i, it has a

core of i which is saved to a node array. After this, the column and row at position j are set to zero. Because we have just updated the degrees of other nodes, we reiterate through the adjacency list and reset the j variable to zero again. At the end of this process we have an array with the core value for every node. The degeneracy of G is the highest core number value of the array. The k-core is the first enriched feature and the only structural feature we are adding to the baseline set. This node property used to detect cliques of human users.

Data: AdjMatrix
Result: CoreArray
i:=0;
while *sum(AdjMatrix)!=0* **do**
  for $j := 0$ **to** $AdjMatrix.length - 1$ **do**
    if $sum(AdjMatrix[j] <= i)$ **then**
      $CoreArray[j] := i$;
      $SetRowZero(AdjMatrix, j)$;
      $SetColumnZero(AdjMatrix, j)$;
      $j := 0$;
    end
  end
  $i := i + 1$
end

Algorithm 1: Algorithm for evaluating core number of a graph.

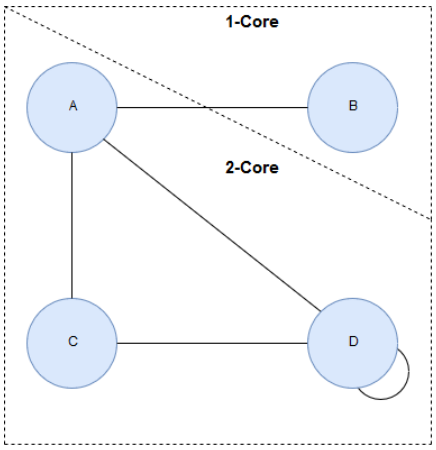


Figure 5: K-core decomposition of Figure 1a graph.

2.5 Machine learning

Machine learning is the prediction of future values through learning and considering past values. Our experiment requires the task of supervised learning. This task is a mapping function for input and output data based on given input and output pair examples. In a classification scenario we used supervised learning to predict whether a given input of a certain class.

Let the set O have n objects with $O = \{o_1, o_2, ..., o_n\}$. Every object o has a specific class of $C = \{c_1, c_2, ..., c_k\}$. In addition, every object o_i, where i is the number of the n-th object, has a set of m features $o_i = \{x_1, x_2, ..., x_m\}$. Machine Learning Classification is a mapping function $y : O \to C$, where we classify an object o with help of the given feature set to a class c. The classifier is trained with a dataset, containing objects and their available classes and from this dataset creates a mapping function. This function can now assign and predict one class of a new object. When training a classifier with features, we must be aware of overfitting or underfitting. Overfitting occurs when the classifier model is too complex, i.e. it has too many features and a small or noisy training dataset. In this situation, the classifier will have difficulty distinguishing between the different classes. Incorrect parameters for the given classification model can lead to overfitting. Underfitting occurs when the classifier fails to model complex datasets.

2.6 Transfer learning

Machine learning trains and predicts information within the same domain. However, when a data source has insufficient information and is too small to train a valuable classifier, it is possible to train from another data source with more information and subsequently apply this to the first, smaller data source. This scenario is called transfer learning. Machine learning generates a prediction in which source, target domain and task are the same. In transfer learning, it is possible to distinguish between these three factors. A definition for general transfer learning task is adopted by Pan et al. [14]:

Definition 6. *(Transfer Learning) Given a source domain D_S and a learning task T_S, a target domain D_T and learning task T_T, transfer learning aims to help improve the learning of the target predictive function $f_T(*)$ in D_T using the knowledge in D_S and T_S, where $D_S \neq D_T$, or $T_S \neq T_T$.*

There are different transfer learning scenarios described in the last sentence of this definition. One describes differing domains of the datasets and one describes a situation in which the learning task is not the same for the source and target dataset. In this thesis, the problem scenario has variable data distribution within the two domains of both source and target datasets. Therefore, another definition is presented below, again adopted by Pan et al. [14], which clarifies the problem of data distribution differences, called transductive transfer learning:

Definition 7. *(Transductive Transfer Learning) Given a source domain D_S and a corresponding learning task T_S, a target domain D_T and a corresponding learning task T_T, transductive transfer learning aims to improve the learning of the target predictive function $f_T(*)$ in D_T using the knowledge in D_S and T_S, where $D_S \neq D_T$ and $T_S = T_T$. In addition, some unlabelled target domain data must be available at training time.*

More precisely, in this thesis we face a different probability distribution within the source and target dataset, $P(D_S) \neq P(D_T)$, in the different *Wiki-Talk* language datasets.

2.7 ROC-AUC

The metric used to measure the performance of the classifier is ROC-AUC score. The term "ROC" stands for "receiver operating characteristic" and is a visual technique for evaluating and selecting classifiers. The ROC displays a graph of the true positive rate (TPR) in contrast with the false positive rate (FPR) [15]. The TPR measures the ratio between true positive classified objects and true positive objects. The FPR measures the ratio between true negative classified objects and real false objects. While plotting the ROC curve, the predicted probability of the TPR is located at the y-axis versus predicted probability. The FPR is on the x-axis. Because we need a single measurement value to evaluate the classifier's performance, we use the "area under the curve (AUC)" value.

2.8 K-fold cross validation

In order to obtain reliable results for a classification prediction, we need to train the classifier. In order to do this, it is not desirable to learn from and test on the entire dataset. Doing this leads to complete conformance of the classes and always results in a nearly 100% prediction accuracy. In this case, the machine learning model only repeats the classification it has learned before. If we try to classify new data, the results will not conform to expectations. To avoid this situation, the dataset is split into a training set and a testing set. The machine learning model trains on the given training dataset and then predicts its learned function for the testing dataset.

For a more reliable evaluation of the out-of-sample data and to use the given training and testing datasets more efficiently, we used the k-fold cross validation method. In this method, the dataset is divided into equally sized K packets . One of these created sets is used for testing and the rest are used for training the model. After learning and testing, the prediction accuracy is calculated. This is repeated K times and in every iteration another K packet is used as a test dataset. Once all iterations are complete, the average accuracy of all rounds is calculated. This value is used as an upper bound baseline, as we assume that accuracy will be lower when testing on out-of-sample data from different datasets or other domains.

The following steps are executed:

1. Divide the dataset into K equally sized packets.
2. One packet is used as a test dataset and the rest as training datasets.
3. The test accuracy is calculated.
4. Repeat step two and three K times, where every iteration uses another package as the testing dataset.
5. Compute the average accuracy as a value for the upper bound baseline for out-of-sample data.

In Figure 6 the procedure is visualised. In the later experiment we calculate the ROC-AUC score instead of the accuracy for step three, and the mean ROC-AUC of all iterations for step five.

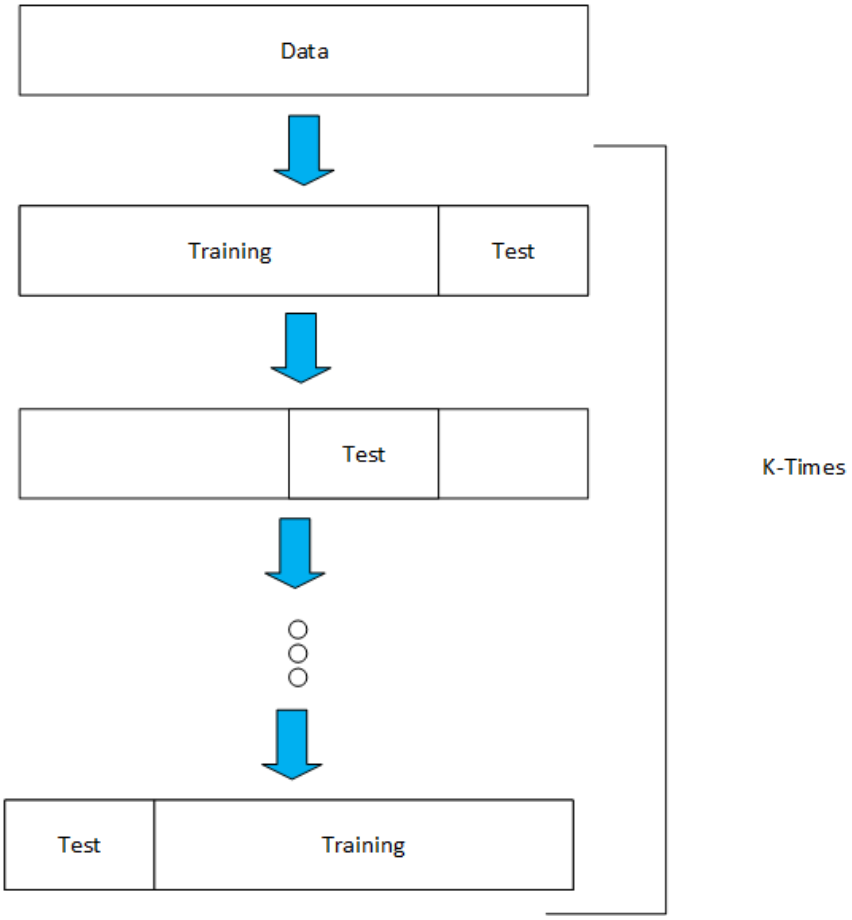


Figure 6: Visualisation of the k-fold cross validation method.

3 Related work

We first introduce the work and research of Jun et al. in this section as the baseline for this work. In the features section we present different research on feature selection for classifying bots with a focus on time evaluation between messages.

3.1 Classification of user roles

Jun et al. [5] present a classification of user roles used in transfer learning . A feature set consisting of structural features was used to classify administrator roles over different networks with additional feature transformation. The goal was to transfer learned knowledge from a source dataset to a feature space common to the target dataset, which had a different data distribution. The feature set used features of degree, indegree, outdegree, local clustering coefficients and PageRank to classify administrator roles. Additionally, the average of the given features from neighbours were added. This method was called "feature aggregation" and all neighbour features were evaluated within a distance of 5 from a given node and feature, and the average of this neighbour features was calculated between every distance and added to the local nodes feature set. The approach attempted to identify administrators with a ROC-AUC score of 0.97 on average using traditional learning and 0.982 on average using the feature transformation approach on 14 languages of the Wiki-Talk dataset. A re-attempt of this experiment predicting all three roles within 27 languages of the Wiki-Talk dataset is presented in Figure 8. These are the scores with proposed settings from Jun et al. [5] with feature aggregation and degree transformation.

3.2 Features

Feature selection is one of the most important elements of training a machine learning model. The given features are responsible for the output classification accuracy. When improperly chosen, the model cannot learn correctly from the given data and ultimately makes incorrect decisions during prediction. In choosing our features in this study, we selected the above mentioned features from Jun et al. [5] because they have already displayed excellent performance in distinguishing administrators from other roles. In addition, we want to add additional structural graph information, i.e. the k-core of a node [12]. With this feature we try to identify clusters of human users that are connected. Research from Welser et al. [16] shows that is possible to classify roles within a network using only the structural signature of a given user node. This feature has the advantage of small computational performance needs compared to computationally intensive content analysis. In [3], Buntain et al. creates a classification of users from the network Reddit[2]. This is accomplished by analysing and using only the network structure and not through the help of any post content. Buntain et al. uses the features of degree distribution and clustering coefficient, which are used by Jun et al. too. These features have already performed well in combination with PageRank in identifying administrators and thus we have retained all of these baseline features in our study.

[2]2 Reddit is a website where users can post content. This can take the form of text or links to other websites. Other users can vote on these posts to rank them on the start page. (https://en.wikipedia.org/wiki/Reddit)

Additional feature selection that is not based on structural properties includes content analysis, various forms of timestamp evaluation and entropy measurements. Since we do not have any information about message content, our approach collects features related to the timing of user messages.

In research by Chu et al. [1], regular and irregular behaviour of users was observed. Chu et al. showed patterns of bot and human activities on a daily and weekly basis. First, they collected a portion of messages and distributed them according to the time of day (by hour) that the message was sent. This was repeated with messages sent on a specific day of the week. They evaluated the regular behaviour for bots within both sets of observations. They found that human users tend to have more irregular activity throughout the day and week. Using this knowledge, Chu et al. created an entropy component in which a low entropy rate indicates a regular process and high entropy indicates an irregular process. This was used to assist the machine learning classifier in distinguishing between human and bot users within the Twitter network.

Wetstone and Nayyar [18] also created a classifier to detect bots in the Twitter network. One approach taken in their work was to create features based on tweet count. One feature interesting for our purposes is the variance in tweet rate, which was calculated based on the number of tweets per hour.

Other Twitter network research from Pozzana and Ferrara [19] studied the behaviour of bots within this network and observed the distribution of time differences between two consecutive tweets. Pozzana and Ferrara found that bots post more messages within one session. Gianvecchio et al. [20] studied the inter-message delay between two messages in internet chats. First, they studied the behaviour of humans and distributed their inter-message delay using a probability mass function. This was then compared with the behaviour of different bot types.

4 Methodology

In this section we present the different methods used in this study. First, we describe how we created the structural features. Here, our timestamp reading method is described and the creation of the new observations based on the datasets for interval, message count and entropy of message proportion are explained. These methods were used for creating various distributions to observe the data and identify any characteristics between the different roles. Feature aggregation and transformation are also presented.

4.1 Feature matrix

The Wiki-Talk datasets consists of two different files for each language. One file contains the timestamps of messages with source and target user identification and the other file, i.e. the group data file, contains the role information of a user. If a message file contains users who are not related to a given role in the group data we assume that this user's role is a normal user. This message file is converted into a directed graph with nodes, where every node represents a user identified by an ID and every directed edge describes one message between two users. From this graph we can create the feature adjacency matrix F with *n* rows , corresponding to nodes and *m* columns for features extracted from the network graph.

$$A = \begin{array}{c} \\ u_1 \\ u_2 \\ . \\ . \\ u_n \end{array} \begin{array}{c} \begin{array}{cccc} f_1 & f_2 & .. & f_m \end{array} \\ \begin{pmatrix} f_1(u_1) & f_2(u_1) & .. & f_m(u_1) \\ f_1(u_2) & f_2(u_2) & .. & f_m(u_2) \\ . & . & . & . \\ . & . & . & . \\ f_1(u_n) & f_2(u_n) & .. & f_m(u_n) \end{pmatrix} \end{array} \tag{8}$$

After creating our basic feature matrix, additional features created with our timestamp feature extraction method are added. For this purpose we process the Wiki-Talk message file again and for every user, u_i, we create a set of his or her message timestamps $M = \{m_1, m_2, .., m_n\}$ in ascending order. Within this timestamp set for each user we create the intervals and count the message number within a specific time frame of day, hour or minute. Different metric functions can now be applied (in this case means and entropy metrics) to create a new feature matrix with timestamp features tf_i. After creating both feature matrices, we append the time to the structural features matrix.

$$B = \begin{array}{c} \\ u_1 \\ u_2 \\ . \\ . \\ u_n \end{array} \begin{array}{c} \begin{array}{cccc} tf_1 & tf_2 & .. & tf_m \end{array} \\ \begin{pmatrix} tf_1(u_1) & tf_2(u_1) & .. & tf_m(u_1) \\ tf_1(u_2) & tf_2(u_2) & .. & tf_m(u_2) \\ . & . & . & . \\ . & . & . & . \\ tf_1(u_n) & tf_2(u_n) & .. & tf_m(u_n) \end{pmatrix} \end{array} \tag{9}$$

4.2 Proportion of messages per weekday and day

A user u_r of role r produces n messages. The function $c_{u_r}(d)$ counts how often u of role r sends a message at weekday d, where d is a day within set
$D = \{Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday\}$. The percentage of all messages by a user of role for a day can be calculated with the function:

$$f_{u_r}(d) = \frac{c_{u_r}(d)}{n} \tag{10}$$

For the observation of all message for a given role we use the function:

$$f_r(d) = \frac{\sum f_{u_r}(d)}{\sum_{d=0}^{6} f_{u_r}(d)} \tag{11}$$

The hourly percentage per role is calculated by counting the messages of a user u_r sent at a specific hour of a day. $c_u(h)$ counts how often u sends a message at a given hour h of a day, represented by a number of the set $H = \{0, 1, 2, .., 22, 23\}$.

$$f_{u_r}(h) = \frac{c_{u_r}(h)}{n} \tag{12}$$

The percentage of all messages for a role is calculated using the function:

$$f_r(h) = \frac{\sum f_{u_r}(h)}{\sum_{h=0}^{23} f_{u_r}(h)} \tag{13}$$

4.3 Entropy of message proportion

To calculate the weekday message entropy, we use the function to calculate the proportion for a given user, equation 10. Here, it is not important which role class the user belongs to, and in later predictions this is also unknown. The function

$$f_{u_i}(d) = \frac{c_{u_i}(d)}{n} \tag{14}$$

calculates the percentage of messages on a given day for a user u_i. The entropy $E(u_i)$ for a single user is

$$E(u_i) = -\sum_{i:c_u(d)\neq 0} f_u(d) log(f_u(d)) \tag{15}$$

Same for the hourly message entropy for a user.

4.4 Intervals

For the later timestamp observation the user intervals are created as follows. Our starting points are role sets $role_i$, where $i <= 2$ Here, 0 indicates a normal user, 1 indicates a bot and 2 indicates an administrator. Any user u within this has his or her own ordered set of timestamps t, where every stamp is the date a message was posted.

$$role_i = \begin{Bmatrix} u_1 = [t_1, .., t_m] \\ . \\ . \\ . \\ u_n = [t_1, .., t_m] \end{Bmatrix} \tag{16}$$

We take every user's set of timestamps and bin the different messages by days. As previously mentioned, the timestamps are sorted in ascending order by time of daye, with $t_n <= t_{n+1}$.

$$user_i = \begin{Bmatrix} d_1 = [t_1, .., t_n] \\ . \\ . \\ . \\ d_n = [t_1, .., t_n] \end{Bmatrix} \tag{17}$$

The intervals between t_1 and t_n can now be calculated for day m, which is the difference of $t_b - t_a$, where $b = a + 1$ and $b < n$:

$$I(d_m) = \begin{bmatrix} t_2 \\ . \\ . \\ . \\ t_n \end{bmatrix} - \begin{bmatrix} t_1 \\ . \\ . \\ . \\ t_{t-1} \end{bmatrix} = \begin{bmatrix} t_2 - t_1 \\ . \\ . \\ . \\ t_n - t_{n-1} \end{bmatrix} \tag{18}$$

The outcome is a set of intervals I for every day a user posted something.

$$role_i = \begin{Bmatrix} u_1 = [I_1, .., I_m] \\ . \\ . \\ . \\ u_n = [I_1, .., I_m] \end{Bmatrix} \tag{19}$$

For the observations of intervals different metrics are applied on the interval arrays. These are the arithmetic mean and geometric mean. The feature subtraction of a given user functions the same way; we do not know which role class the specific user belongs to and we begin with Equation 17 and evaluate the different time delays between messages. Finally, we apply a given metric on the set of intervals for each user to create a feature.

4.5 Feature aggregation

The feature aggregation method of Jun et al. [5] is a collection of average feature values from a local node's neighbourhood. A maximal depth of r is given, which indicates how many jumps should be made from a start node to collect neighbour information. For every jump the information of a given node for all neighbours within the range of r is collected, and the average is calculated and added as a new feature to the local start node. In the later experiment r is set to 5, as proposed by Jun et al., because real networks have a small diameter. It should be noted that the feature set increases by the number of given features with every r. This has high computational demands, first in computing the aggregated features and later in the prediction of users.

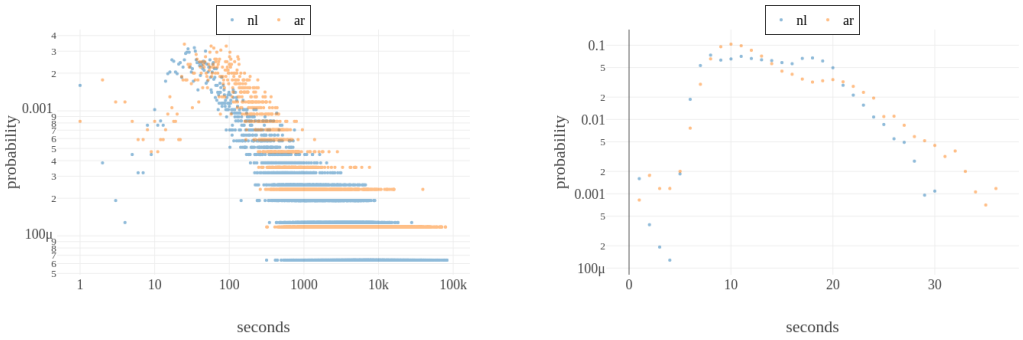
4.6 Feature transformation

When comparing two datasets two different data distributions are considered. In our case, the datasets have different degrees and interval distribution s. When we learn from a dataset and want to apply this learned knowledge to a different dataset, the classifier faces difficulties due to the different domains of the source and target dataset and future predictions are incorrect. The data of both sets cannot be used directly, and we have to transform them to a common feature space. There are different methods available for transforming features to a common feature space. A simple normalization method is presented by Saputra et al. [21]

$$\hat{x} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{20}$$

Here, $\hat{x}$ is the normalized value of a set X. x_{min} and x_{max} are the minimum and maximum of the given set. This normalization method can be applied on the average message rate of a given user.

Another approach to feature transformation is the transformation of distributions following the power law, where the constant of a power distribution is shifted for the source and target dataset to a common power law distribution [22]. An implementation of this transformation is presented in the work of Jun et al. [5]. We utilize this approach within the features of degree, k-core, intervals and message rate. All three distributions follow the power law. An example of the interval transformation is presented in Figure 7.



(a) Non-transformed

(b) Transformed

Figure 7: Average interval by user distribution of datasets *nl* and *ar*. Before and after power law transformation.

5 Experiments

The experiment is divided into different parts. In this chapter, we first provide an introduction to the experiment's implementation. The baseline score is introduced using results from the work of Jun et al. [5], which we use as a lower bound for improvement in this study. This chapter continues with the observation, which provides answers to our research questions. First, we observe the dataset with regard to the activity of roles over a period of years. Here we also examine the message distribution between the roles and explain why the administrator classifier for the recent work reaches the given scores presented in Figure 8. In the second half of this chapter we present the timestamp observation, including the interval frequencies of two continuous messages distributed by role. To observe the user's behaviour of sending messages in series with short delays, we apply different mean functions to detect the average time between these delays. We analyse message rates within a specific time frame of rate per minute, hour and day. Finally, an observation of noticeable regularities or irregularities in the behaviour of message sending was conducted by evaluating the distribution of messages sent at a given time of a day. We also examined the distribution of messages sent on a given day of the week. With this data we are able answer our three main research questions regarding regular behaviour, noticeable interval length and message rates for bots.

5.1 Implementation

Baseline codes for reading the structural information of the datasets and transfer learning between different sets were created by Jun et al. [5]. Our experiment code was also written in Python and consists of the machine learning element (to evaluate scores with single enriched features), timestamp evaluation for a role, extract timestamp features for single users and a transfer learning element for testing the final feature set. We have created a method to read the timestamps from each user message and bin these in one day buckets. Thus, we can make calculations and observations regarding these daily messages. The different plotted observations are created with the data visualisation tool plotly [24]. Machine and transfer learning results are also plotted with this tool. The learning and predictions are generated with the Random Forest Machine Learning Classifier, provided by sklearn [25].

5.2 Baselines

For predicting bots within the Wiki-Talk datasets we use a Random Forest Classifier, which is trained and tested with the features degree, indegree, outdegree, local clustering coefficient and PageRank. As an additional feature the average values of neighbours are added, with a maximal neighbour depth of 5. Results for the repetition of the experiment by Jun et al. [5] for all three roles are presented in Figure 8. The administrator classifier performance the best, right after the user and with the least performance the bot classifier . Based on these scores, the experiment in this thesis was designed to focus on the improvement of predicting bots.

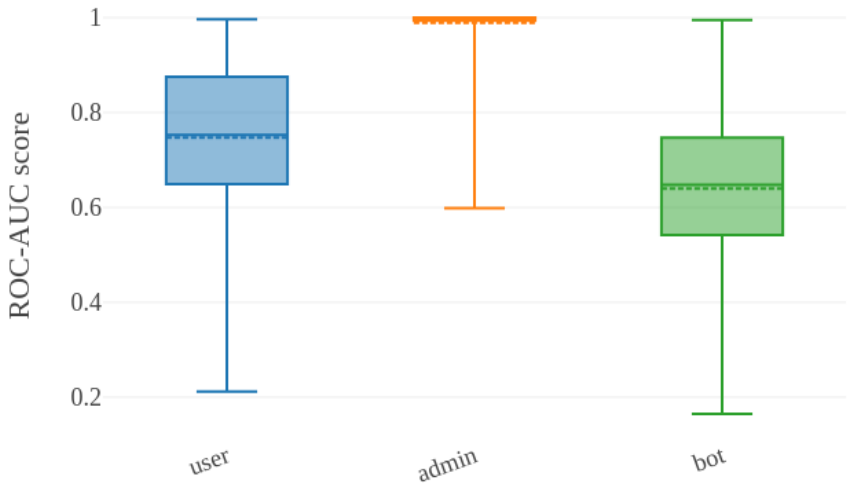
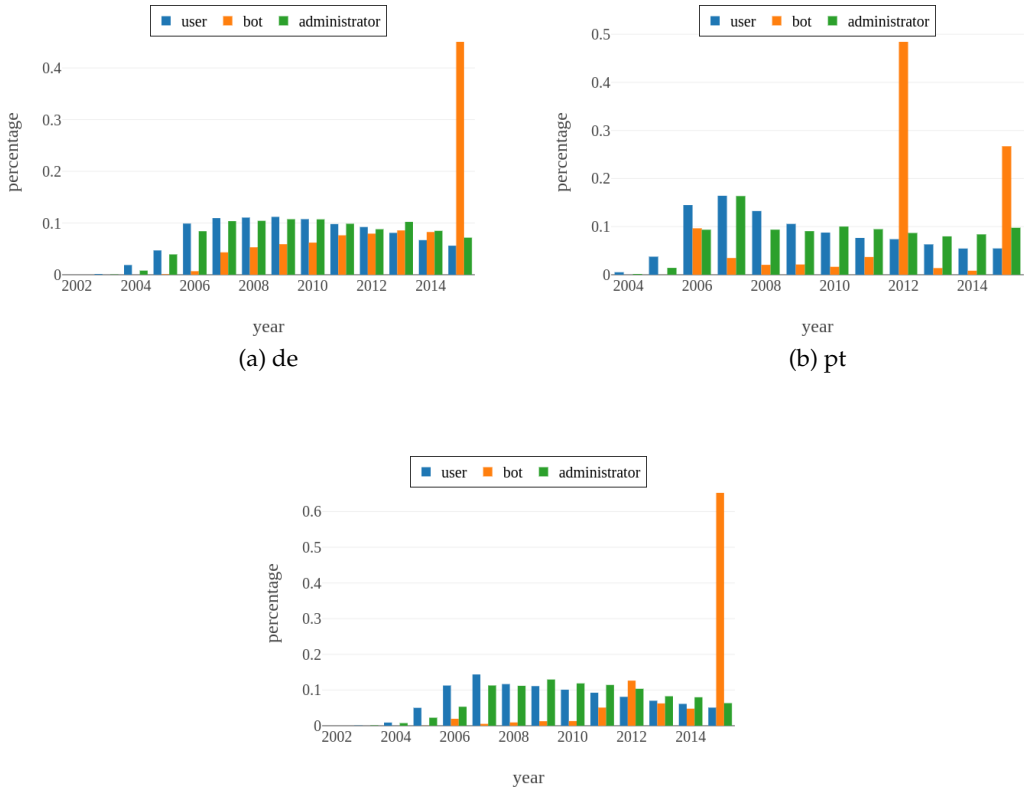


Figure 8: ROC-AUC scores of predicting administrators, bots and normal users in all sub-datasets. The dotted line is the mean of all scores for the given dataset as the learning set, targeting all other datasets. The continuous line within each box represents the median and the dotted line represents the mean of a given sample.

Role	Mean	Median	Min	Max
user	0.7471	0.7524	0.2116	0.9961
admin	0.989	0.9991	0.5985	1.0
bot	0.6396	0.6478	0.1648	0.995

Table 2: Baseline transfer learning metric scores.

5.3 Dataset observation



(a) de

(b) pt

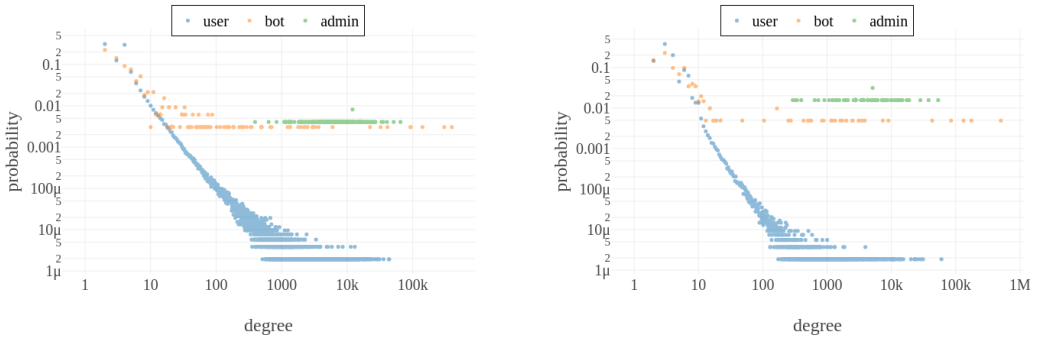
(c) pl

Figure 9: Distribution of message activity from all available messages within a set. Language datasets *de*, *pt* and *pl* are shown.

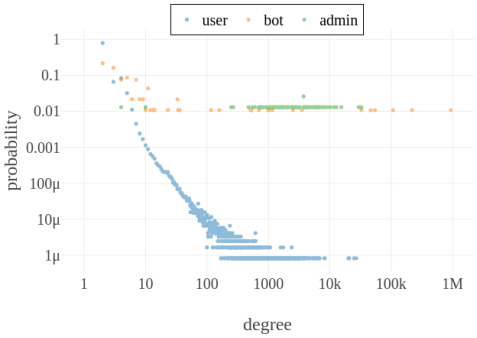
In Figure 9 the message activity for each role is shown for languages *de* and *pt*. High activity for bots is seen in most datasets for the most recent years of data collection. Some sets have more activity the years before, such as set *de* in Figure 9a, but predominantly, high participation is seen for bots in the most recent years represented in the datasets . For example, language *pt* (Figure 9b) has nearly 50% of all its bot activity in the year 2012. Another extreme case is seen for the language *pl* (Figure 9c), for which over 60% of all bot activity takes place within the year 2015. This behaviour can be observed throughout all sets, from Figure 32a to 32aa, presented in appendix. One reason for these high activity levels in the most recent time periods could be the more common usage and development of bots in the last decade. Also, a change in policy by Wikipedia may have allowed a more common usage of bots.

We use the complete dataset of a language for observations in this section. Other approaches based on timestamp observation of inter-message delay between two

messages from the same user utilized a smaller time frame for observation. The research of Gianvecchio et al. [20], for example, considers only one month, since this research was based on Twitter datasets where a high volume of information and messages exist within the time span of one month. From the activity time observation and Chu's [1], of evaluating the account registration dates, we take the activity duration of users for an additional time feature. Because we do not have information from registration dates, we use the time between first and last message posted by a given user as the activity time duration.
Figure 10 presents the degree distribution between different roles. The degree is the sum of incoming and outgoing messages from a user. The distribution of normal users follows the power law, with a high number of low degrees and a long tail of lesser users with high degree numbers. Bots have both small and high degrees, from 1 to the 100,000 mark, with some outliers. The administrator role has a continuous line of distribution within a high range of degree at the same level of probability occurrences , beginning at 100 and reaching 100,000. In Table 8 (see appendix), we listed the average degrees by role for every dataset. In all datasets normal users show a clear difference in their average degree compared to other roles, small overall within the range of 1.65 for dataset *bn* and 17.38 for dataset *br*. Administrators tend to have a high average degree, distinct from normal users. In some datasets, like *oc*, bots have a small but also a high average such as in dataset *zh* with $15,155.01$. The language *pl*, for example, has an average degree of $4,970$, which is nearly the average of its administrator role at $4,755.0$. This could be related to outliers, observing *zh* in Figure 10c with one outlier nearly by a degree of one million. With this observation the structural features, based on degree, perform well in identifying administrators by their high degree compared to the two other roles. Due their degree, however, bots can be identified as normal users or administrators. Figure 11 presents the k-core distribution. Again here, the same behaviour like the degree distribution can be observed. High core numbers for administrators and the bot and normal user roles are present in the complete range.

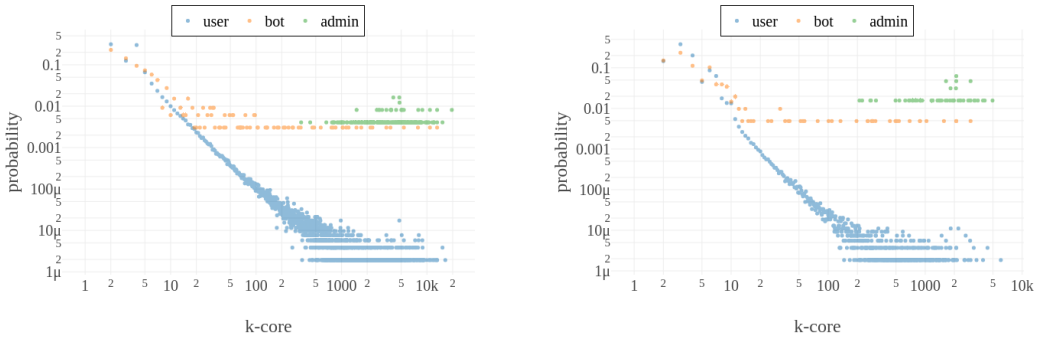


(a) de (b) pt



(c) zh

Figure 10: Degree distribution by role for datasets *de*, *pt* and *zh*.



(a) de (b) pt

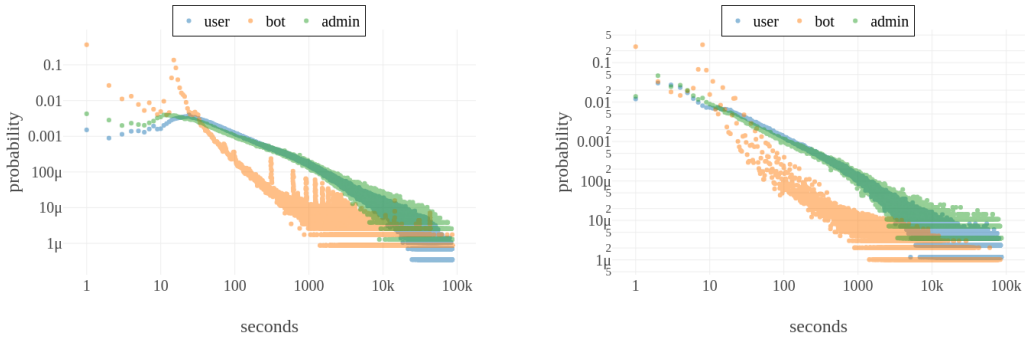
Figure 11: K-core distribution by role for datasets *de* and *pt*.

5.4 Evaluation of timestamps

The baseline feature set consists of features that correlate to the degree of a node. The in- and outdegrees are clearly related to the degree. The local clustering coefficient measures how a node's neighbours form a clique or cluster. This is dependent on the node and its neighbour's degree. PageRank calculates the importance of a node, also according to its degree, and its linked nodes. Although we do not have information regarding the content of the different messages, we do have information about the given date and time when a specific message was sent. With this information we can make different observations about the roles and their message time behaviour and answer our research questions.
Inspired by the work of Pozzana et al. [19], who propose observing and using time differences between two sequential messages, as well as by Gianvecchio et al. [20] who suggest observing the inter-message delay between two messages to classify bot users in chat networks, we assume that bots have a short time delay between their different messages. In the following section we use this time difference between two consecutive messages. The mean of users' intervals is also presented, which serves as a feature in the classification.
We also want to observe the message count for specific roles within different time frames and calculate the mean message count per time frame [18]. Wetstone and Nayyar proposed an hourly message rate within the Twitter domain. Because we assume that bot users tend to send many messages within a small time frame, we track the message rate within 1 minute. It must be noted that we have different user behaviour than seen for Twitter, with much fewer messages per user. With this in mind we also observe the message rate over the course of a day. Another assumption from the work of Chu et al. [1] is to track the message behaviour on a daily and weekly basis to identify irregular and regular user behaviour. Based on this research we assume that human Wikipedia users exhibit irregular behaviour, which helps distinguish them from bots.

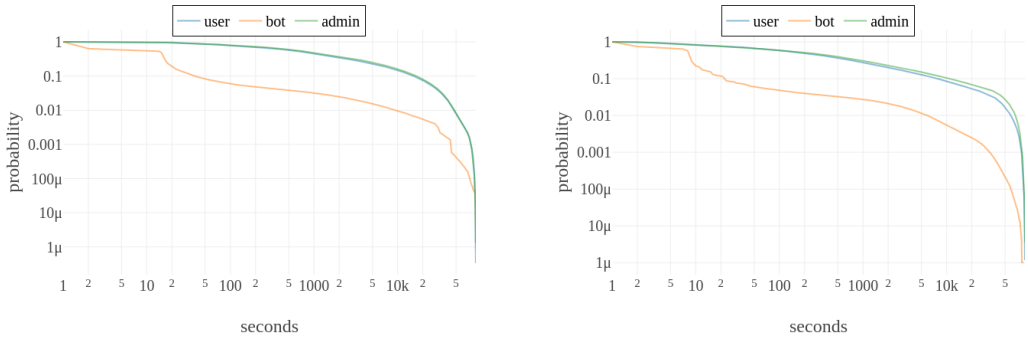
In the next section all intervals are shifted by a value of 1.0. Because graphs are shown with logarithmic axes, we wish to note the high occurrences of the 0.0 seconds interval threw the different roles.

5.4.1 Interval between messages



(a) de (b) pt

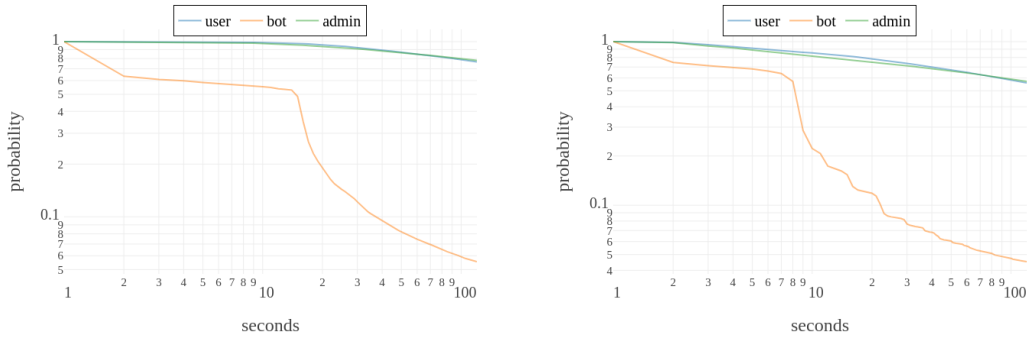
Figure 12: Interval distribution per day for language dataset *de* and *pt*. All intervals between messages for each user from a specific role are counted.



(a) de (b) pt

Figure 13: The complementary cumulative distribution function of the intervals in seconds.

Using the approach of observing the time differences between two consecutive tweets from Pozzana et al. [19] and distribution of human inter-message delay from Gianvecchio et al. [20], we assume that bots can send messages within a short time period. Computer controlled bots can respond to messages more quickly than humans and can execute more activities within a shorter time period. An observation of time intervals between messages can therefore be useful. To obtain an initial sense of how the intervals are distributed we created and observed frequency graphs, beginning with an overview of the delay between messages per day.



(a) de (b) pt

Figure 14: The complementary cumulative distribution function of the intervals within a time frame of 120 seconds.

Figure 12 present the time intervals for datasets *pt* and *de*. We calculate the differences between two continuous messages within each day and count how often each interval appears for each role. All intervals from each user are calculated and every marker represents one interval. On the x-axis the interval length is shown in seconds. On the y-axis the probability of its occurrence is shown. Across all datasets the high number of intervals with 1 second can be observed. Users in the bot role class have a high probability (over 90%) of exhibiting these delays. It should be noted, however, that normal users and administrators also have a proportion of intervals of 1 second. These 1 second delays can be automatic responses from mechanisms associated with user accounts within the Wikipedia network. Another reason for these delays could be a new feature or policy change within the network, which is not signified through mining the datasets. One function could be that users run scripts from their accounts. In general scripts are separated to another account, which are flagged as bot accounts. When investigating the message there is no clear pattern to observe threw the different timestamps, where the 1 second intervals occur. Seen in Table 3, different messages from users where the scenario appears. Users with ID 145, 423 and 500 send two messages at the same time to their self, with a delay of 0 seconds. This can be an indicator for an automated script.

The complementary cumulative distribution function is shown in figure 13. The curve of normal user and administrator roles behaves the same as observed above in the frequency distribution. We also see an increase in the slope from bots after intervals of over 10 seconds. A strong slope can be observed for bot roles after the 10 second mark for dataset *de* and before the 10 second mark for *pt*. In dataset *de* there is a probability of under 10% that a bot user will display intervals of over 40 seconds, while for *pt* this interval is around 20 seconds.

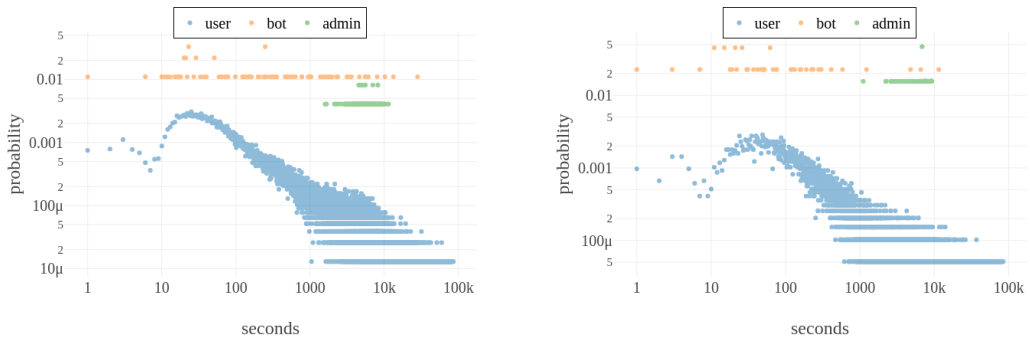
To obtain a better overview of what happens within smaller interval distributions, we examine intervals that are shorter than 2 minutes in Figure 14. Here, we can

User ID	Role	Zero Interval Messages
61	User	'2006-12-10T12:48:44Z', Target ID: 75, Admin; '2006-12-10T12:48:44Z', Target ID: 45, Admin
145	User	'2010-01-27T21:59:32Z', Target ID: 145, User; '2010-01-27T21:59:32Z', Target ID: 145, User
764	User	'2014-09-28T15:25:29Z', Target ID: 26986, User; '2014-09-28T15:25:29Z', Target ID: 32654, User
11016	User	'2012-02-18T17:34:19Z', Target ID: 8277, User; '2012-02-18T17:34:19Z', Target ID: 20082, User
9425	User	'2015-08-14T03:07:49Z', Target ID: 9425, User; '2015-08-14T03:07:49Z', Target ID: 16827, Bot
423	Admin	'2011-07-12T13:10:52Z', Target ID: 423, Admin; '2011-07-12T13:10:52Z', Target ID: 423, Admin
500	Admin	'2011-03-06T18:50:11Z', Target ID: 500, Admin; '2011-03-06T18:50:11Z', Target ID: 500, Admin

Table 3: Example for users with two continues messages with 0 second interval. Shown are the time stamp of the messages with the ID and role from the recipient of the message. Examples are taken from the language dataset *br*.

observe a step from the 1 to the 2 second mark. After this step, with a probability of 40% for dataset *de* and 30% for *pt*, the intervals do not become smaller than 2 seconds. The next decline is between the 10 and 20 second mark for *de* and after the 7 second mark for *pt*. The administrator and normal user roles behave the same, exhibiting a slow slope with a small gap between the curves within the *pt* dataset. From this observation we can answer our second research question affirmatively, as there are a noticeable number of short intervals and an especially high number with the value of 1 second for users in bot roles.

5.4.2 Arithmetic mean of intervals per user

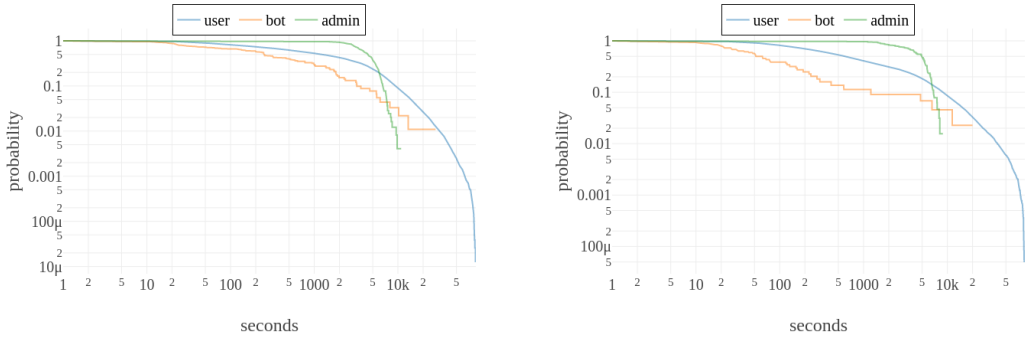


(a) de (b) pl

Figure 15: Average interval distribution per day. Each marker in the graph presents the arithmetic mean of a user's interval set.

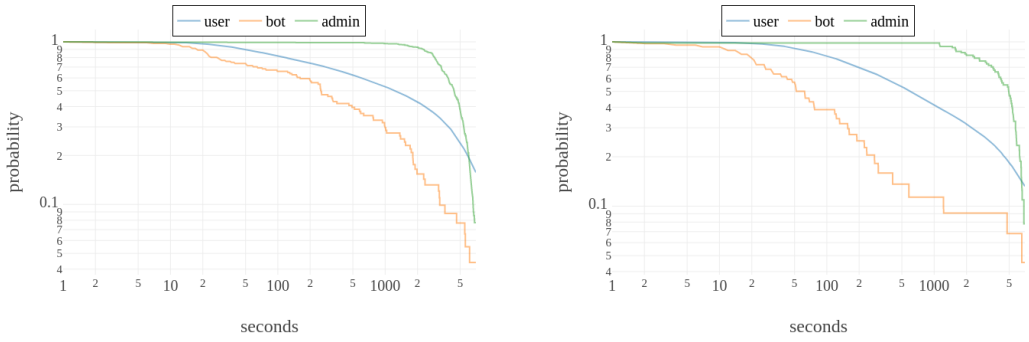
From the above observations we know that all roles have a proportion of short intervals between messages, especially bots and the normal user role. Following we calculate the average of daily intervals for every user, shown in Figure 15. Here, there are two noticeable observations. First, the average interval for normal users has a distribution that follows the power law with some outliers in the lower interval lengths. Second, bots and administrator users have a continuous horizontal line of distribution. This is the result of the low number of occurrences of users within this role. Administrators are primary present in the larger average intervals, after $1,000$ seconds. Before this mark they are not present at all. Bots and normal users are present in all interval ranges.

The complementary cumulative distribution function graph for the average interval per user is shown in 16. All three roles exhibit the same slope behaviour for intervals of up to 10 seconds. After this point, the curves separate, with the steepest slope for bots, then normal users and finally administrators. A strong decreasing curve can be observed after around $2,000$ seconds for administrators. In this graph we specifically want to see what happens when the curves separate. The complementary cumulative distribution function graphs in Figure 17 show the curves within intervals smaller than $6,000$ seconds. Here we can see that the gap between normal users and bots is smaller than in the complementary cumulative distribution function graphs of Figure 14. Nevertheless, they separate early after 10 seconds and the probability that values will become larger than 10 seconds is over 90% for the normal user and administrator roles. Based on this observation of the average interval of a user class, adding a feature with this information to the classifier helps distinguish bots and normal users from administrators. However, there is no clear differentiation between bots and normal users.



(a) de (b) pt

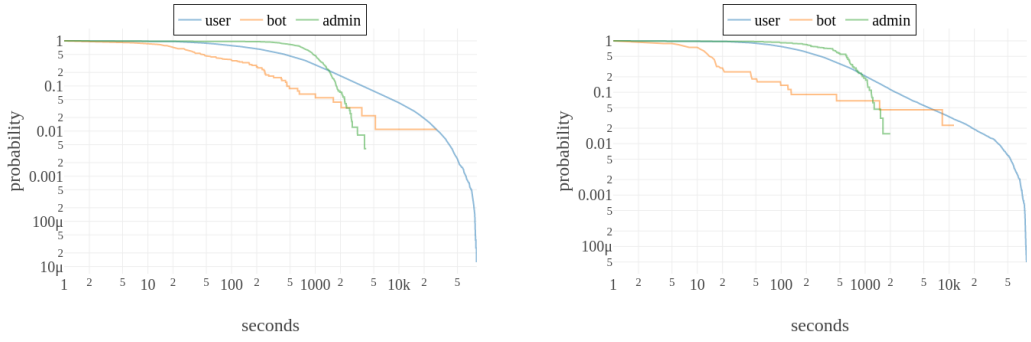
Figure 16: Complementary cumulative distribution function of intervals.



(a) de (b) pt

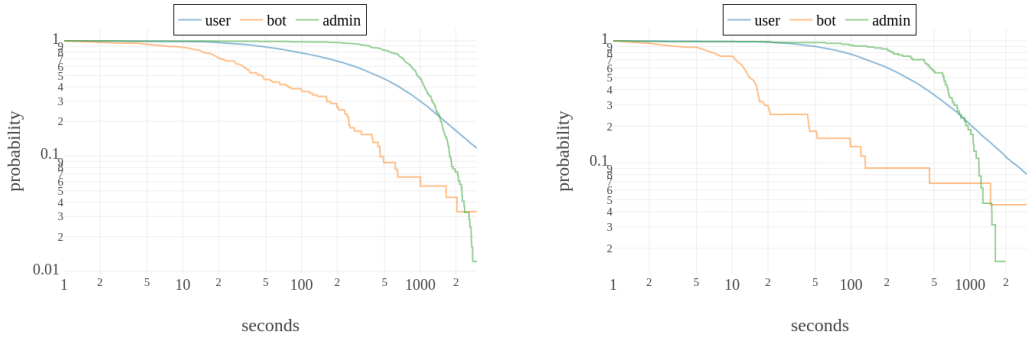
Figure 17: Complementary cumulative distribution function of intervals under $6,000$ seconds.

5.4.3 Geometric mean of intervals per user



(a) de (b) pt

Figure 18: Complementary cumulative distribution function graph of the geometrical mean of intervals between messages.



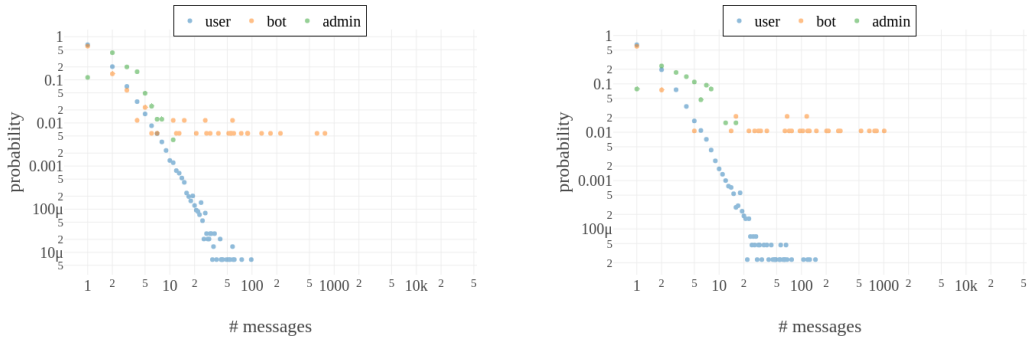
(a) de (b) pt

Figure 19: Complementary cumulative distribution function graph of the geometrical mean from intervals between messages under $3,000$ seconds length.

Plotting the geometrical mean of user intervals provides a clearer differentiation between humans and bots . The graphs in Figure 18 show that human roles exhibit the same behaviour within intervals that are under 20 seconds. The wide gap between bots and humans is noticeable here. When considering a smaller section of the graph with a maximum time frame of $3,000$ second intervals, it can be seen that the length of this gap between bots and humans increases after intervals of over 20 seconds for *de* and after intervals of over 10 seconds for *pt*. In dataset *de*, bots have less than a 10% probability of having a mean interval value of over 50 seconds,

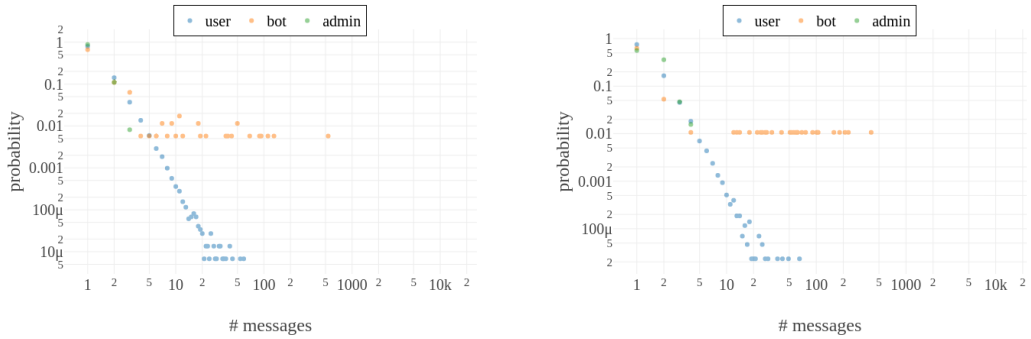
whereas a normal user's probability of exhibiting this interval value is around 50%. In dataset *pt*, bots have less than a 10% probability of having a mean interval value of over 120 seconds, with a normal user's probability resting at around 70%. The administrator role is above both slopes and decreases rapidly at the 500 second mark for both datasets. Compared to the arithmetic means in 19, we got a higher probability to identify bot users with a low geometrical mean interval than a normal users.

5.4.4 Arithmetic mean of message rate per time frame



(a) de (b) pt

Figure 20: Average user message count per day.

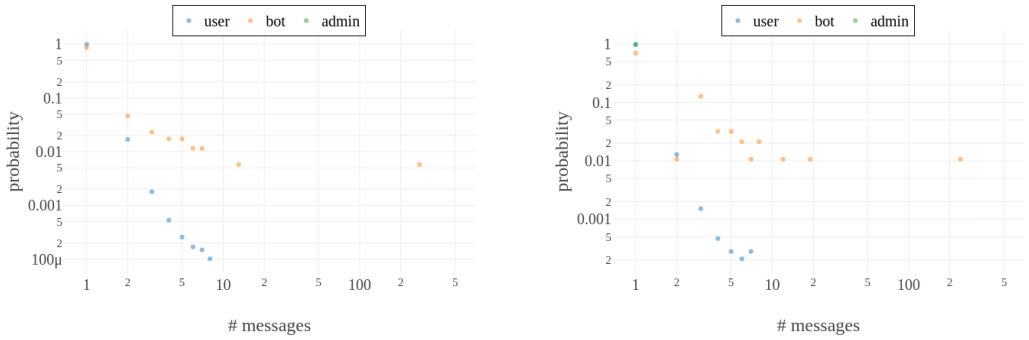


(a) de (b) pt

Figure 21: Average user message count per hour.

Inspired by the paper from Wetstone and Nayyar [18] we decided to evaluate average message rates within a specific time frame. The approach was to count the number of Tweets per hour when the user was active. Since we are working with a social network domain that does not have as high a frequency as Twitter, we also considered average daily message rates.

First, we examined the average number of messages per day to determine whether there were any differences between the number of messages across the three roles. In Figure 20 the average message rate for datasets de and pt are shown. Normal users and bots have a high frequency of only one message on average per day. The distribution of normal users follows the power law, with a small tail, whereas the bot



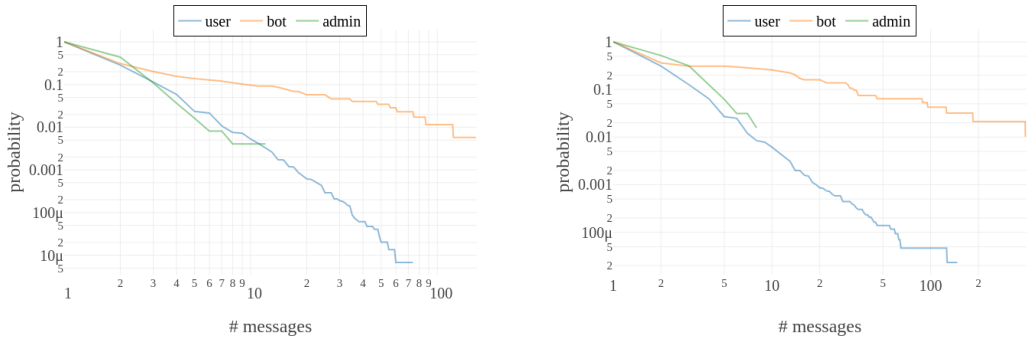
(a) de (b) pt

Figure 22: Average user message count per minute.

users tend to exhibit a larger tail. Normal users have a message count distribution of up to 100 messages per day in dataset *de* and up to 200 in dataset *pt*. Administrators have a maximum volume of 20 messages per day. Examining the bot role, we see a line of message rates with the same probability and a maximum of nearly $1,000$ messages for both datasets. The message rate distribution appears the same when we view the hourly message rates. Again, all three roles have a high probability of just one message within this time frame. Within the rates of messages per minute we can observe that administrators are not present for dataset *de* and exhibit a rate of 1 for dataset *pt*. Bots are an outlier with a high rate, and they have a higher probability than normal users of having message rates greater than 1. The average rate of messages can only function as an indicator for classifying differences between bots and the two human roles when we have a bot with high messages rates. In the low message rate cases, this predictor fails because of the high number of low rates for all roles. Only the minute time frame with high bot activity assists in classification.

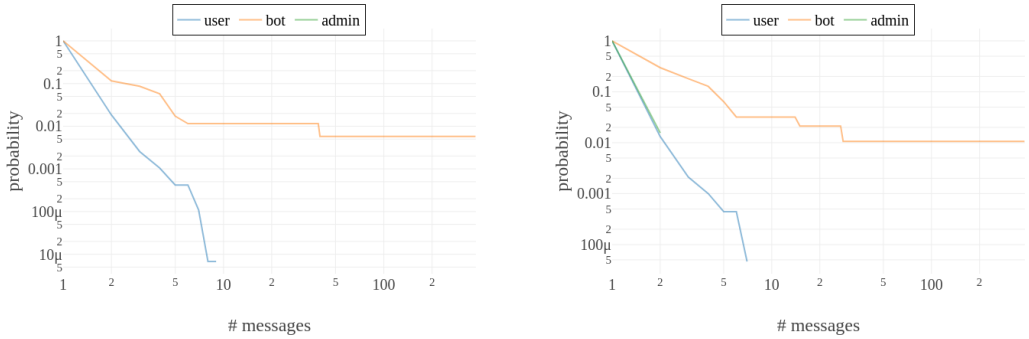
5.4.5 Geometric mean of message rate per time frame

Because we already know the arithmetic distribution of rates within the different time frames, we are interested here only in the complementary cumulative distribution functions of the geometric mean because we observe the same phenomena as in the normal average rate distribution. Within daily rates in Figure 23, we see a distinction after two or three messages, where humans tend to have lower rates and bots tend to have higher rates. In the hourly rates, Figure 24, this behaviour increases, and the administrator role exhibits no rates higher than two messages per hour. Within the minute-based complementary cumulative distribution function, Figure 25 we have a better understanding of what happens. Again, administrators are only present with a maximum rate of two messages in the *pt* dataset present and not in *de*. We can also observe that bots have a constant probability of having message rates between 6 and 40 for dataset *de* and a $1,000$ times higher probability of having users with more than eight messages per minute.



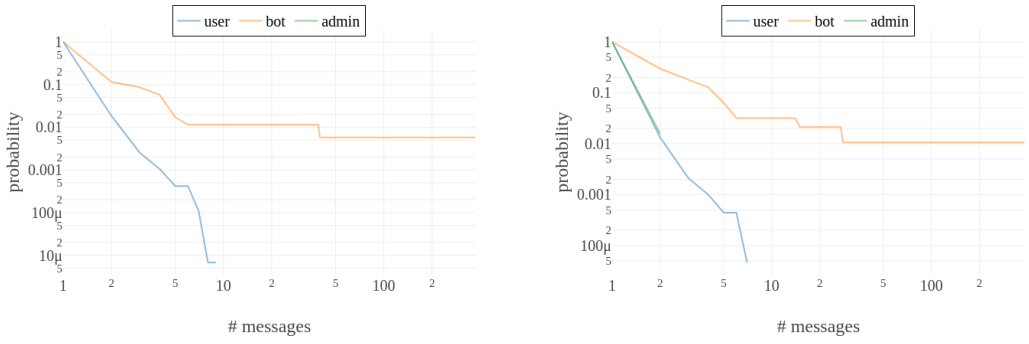
(a) de (b) pt

Figure 23: Complementary cumulative distribution function graph of the geometrical mean from user message rates per day.



(a) de (b) pt

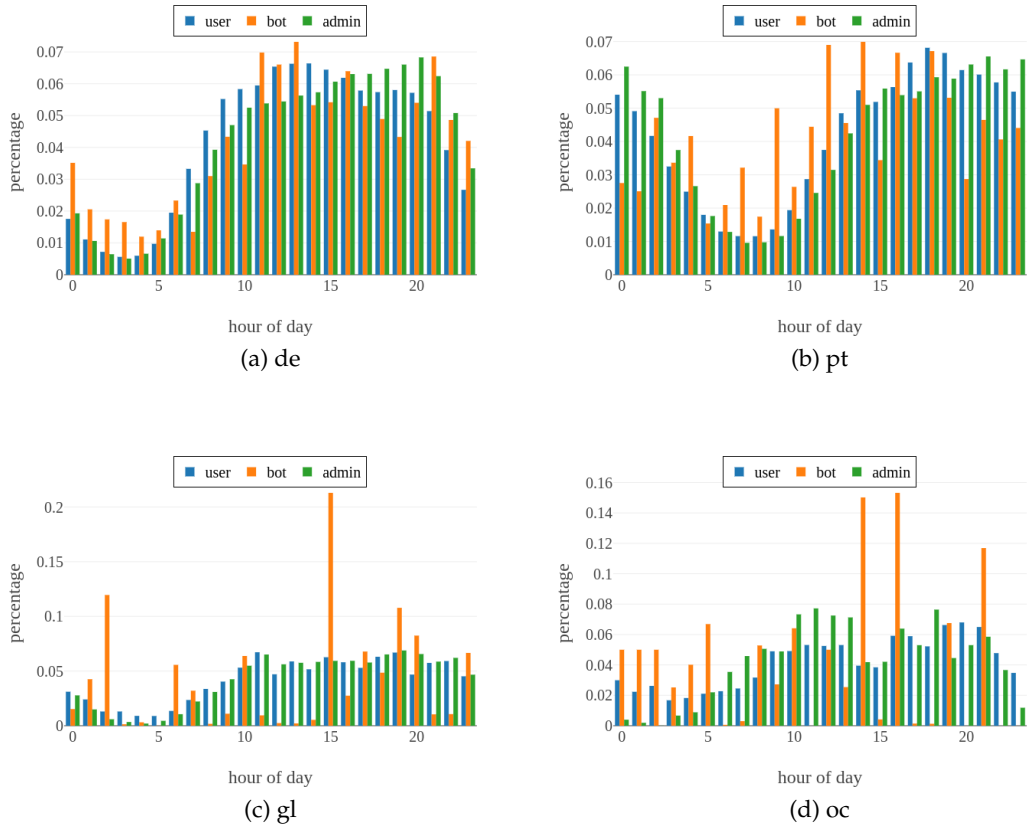
Figure 24: Complementary cumulative distribution function graph of the geometrical mean from user message rates per hour.



(a) de (b) pt

Figure 25: Complementary cumulative distribution function graph of the geometrical mean from user message rates per minute.

5.4.6 Day and weekday message distribution

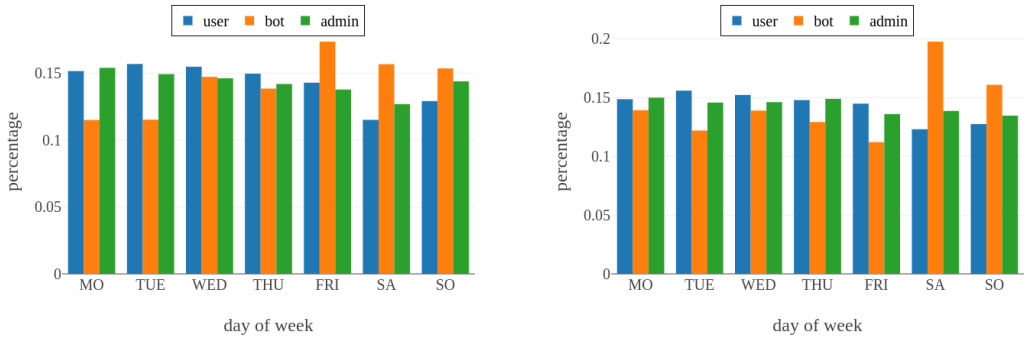


(a) de (b) pt

(c) gl (d) oc

Figure 26: Daily message distribution by role. User posts are tracked by specific hour of the day.

The last observation is the daily message behaviour of users. We assume that bots exhibit more regular behaviour than human users [1]. In Figure 26 the daily message distribution of users is shown for the sub-datasets *de* and *pt*. Here, the hour of the day when a message was sent by users of a specific role class is presented. Our assumption is shown to be incorrect, and it can be observed that humans users, i.e. normal users and administrators, tend to display regular message behaviour, contradicting to our assertion. In dataset *de* a more regular behaviour for bots can be seen, but other dataset graphs show some irregularities in bot messaging, such as that observed in datasets *pt*, *gl* and *oc*. The high peaks for bot messaging at certain times of day are noticeable. The second proposed method for time observation, offered by Chu [1], is to evaluate message count for a given day of the week. This is shown in Figure 27. Across all datasets this distribution is regular for all roles. A message count entropy feature for the day of the week does not appear to be a



(a) de (b) pt

Figure 27: Weekday message distribution by role.

reliable indicator for identifying bots.

6 Evaluation

In this section we test our observed features individually with the baseline feature set. After testing we create our final set for transfer learning and evaluate the scores for this learning and knowledge application with different feature transformation configurations.

6.1 Feature evaluation

From the observations we identified three different categories of new features: message rate, interval and the category of other features. Tested are also the means for maximum and minimum interval from a user. These two additional features are in the category interval features. We added the different characteristics individually to the feature set and tested them with a 10-fold cross validation within each language dataset. We chose 10 folds because sub-dataset *gl* has only 12 bots and the least populated class requires k members for a valid k-fold cross validation. For every fold we calculated the ROC-AUC score. After 10 folds were complete, the mean of all scores was calculated. This helped identify the correct features and avoid adding features all at once with the risk of overfitting the classifier. An overview of ROC-AUC scores for evaluation with the baseline feature set can be seen in Figure 48, with specific scores in Table 9 listed in the appendix.

Scores for count features are presented in figure 28. Dots in the plot present the baseline scores with minimum and maximum scores as the blue dots, median as the black dot and the average as red dot. The feature arithmetic mean of messages per minute is taken to enrich the feature set in the transfer learning. It scores the highest maximal of all features, better performance in the median, mean and minimum than the baseline scores and the other message rate features. From the interval features in Figure 29 the geometric mean of intervals from every user's interval set are taken as features. For each feature we see a higher minimum and maximum score than observed in the baseline. From the last group of features in 30 only the k-core feature are taken, through the raise in the minimum score. Finally, three new features are added to the baseline feature set. These are:

- arithmetic mean of message rates per minute
- geometric mean of intervals
- k-core

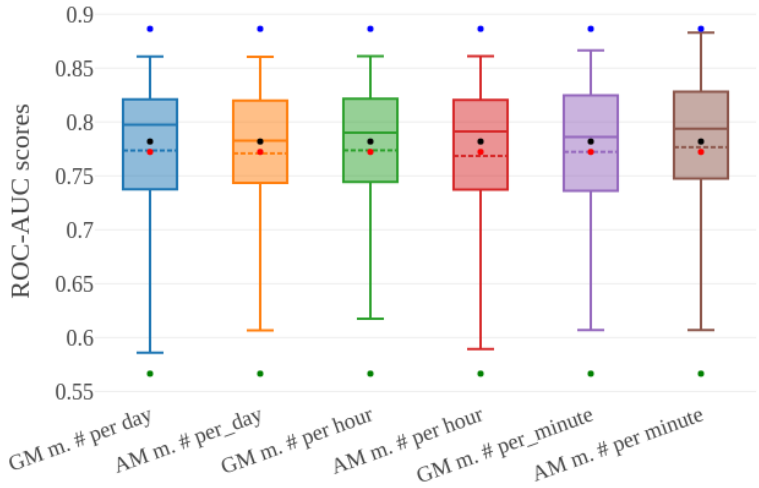


Figure 28: Overview of ROC-AUC scores with arithmetic and geometric message rates.

Feature	**Min**	**Max**	**Median**	**Mean**
GM m. # per day	0.5859	0.8608	0.7974	0.7736
AM m. # per day	0.6067	0.8606	0.7826	0.7709
GM m. # per hour	0.6174	0.8611	0.79	0.7737
AM m. # per hour	0.5895	0.8611	0.7912	0.7686
GM m. # per minute	0.607	0.8664	0.7862	0.7723
AM m. # per minute	0.6069	0.8831	0.7937	0.7767
Baseline	0.5665	0.8865	0.7819	0.7722

Table 4: ROC-AUC scores for message rate features.

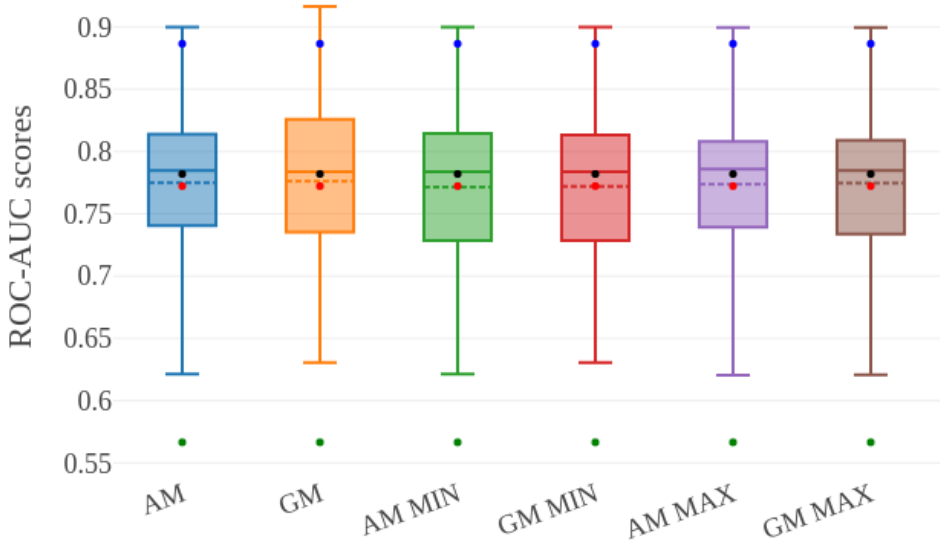


Figure 29: Overview of ROC-AUC score with arithmetic and geometric intervals, minimum and maximum of a users interval.

Feature	**Min**	**Max**	**Median**	**Mean**
AM	0.6213	0.8997	0.7847	0.7749
GM	0.6304	0.9164	0.7837	0.7761
AM MIN	0.6214	0.8997	0.7837	0.7714
GM MIN	0.6304	0.8997	0.7837	0.772
AM MAX	0.6205	0.8995	0.786	0.7737
GM MAX	0.6208	0.8995	0.7845	0.7746
Baseline	0.5665	0.8865	0.7819	0.7722

Table 5: ROC-AUC scores for interval features.

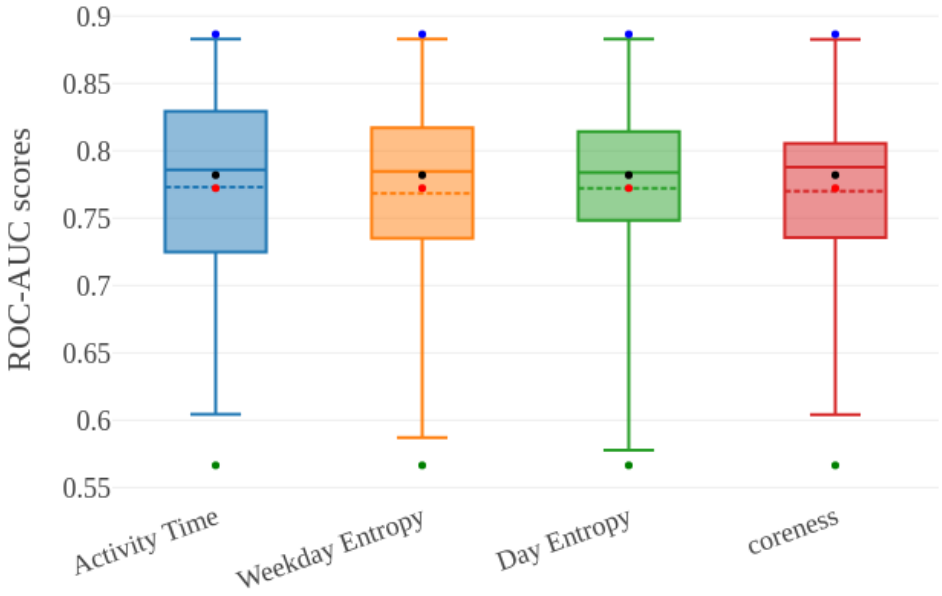


Figure 30: ROC-AUC score with different features.

Feature	**Min**	**Max**	**Median**	**Mean**
Activity Time	0.6043	0.8828	0.7859	0.773
Day Entropy	0.5777	0.8828	0.7838	0.7722
Weekday Entropy	0.587	0.883	0.7848	0.7683
Coreness	0.6042	0.8828	0.7879	0.77
Baseline	0.5665	0.8865	0.7819	0.7722

Table 6: ROC-AUC scores for different features.

6.2 Transfer learning results

The transfer learning was tested with the feature set of degree, in- and outdegree, PageRank, local clustering coefficient, k-core and the time features of average message number per minute and geometric mean of user intervals. In Figure 31the transfer learning performance is presented with various transformation configurations. Each bar represents the score of learning from each language set and the prediction of bots within all other sets. From 27 languages we have a total of 729 learn and predict combinations. The transformation methods are power law transformation, labelled as “trans”, where all degree, k-core and timestamp features are transformed; normalization method on the average message rate per minute, labelled as “norm”, with degree and interval power law transformation; and degree transformation alone, with no other transformation and labelled as "d. trans". The label baseline and enriched are scores without any transformation, where “baseline” means the feature set of degree, indegree, outdegree, local clustering coefficient and PageRank and enriched, the five features from the previous mentioned baseline features with geometric mean of interval, average message number per minute and k-core.

Compared to the non-transformed scores, the enriched feature set improves the mean, minimum and maximum scores with respect to the baseline. The best scores with transformed features are reached with power law transformation of degree and interval and the normalization of message rates. An improvement of 0.61% in mean and 0.24% in the median was achieved with a simultaneous 3.23% decrease in the minimum.

Feature	**Mean**	**Median**	**Min**	**Max**
baseline	0.6218	0.6304	0.1061	0.9552
enriched	0.6291	0.6312	0.1455	0.9968
baseline trans.	0.6396	0.6478	0.1648	0.995
enriched d. trans	0.6457	0.6502	0.1401	0.993
enriched trans.	0.647	0.6518	0.1325	0.9929
enriched trans. norm.	0.6457	0.6478	0.1329	0.993

Table 7: Transfer learning metric scores.

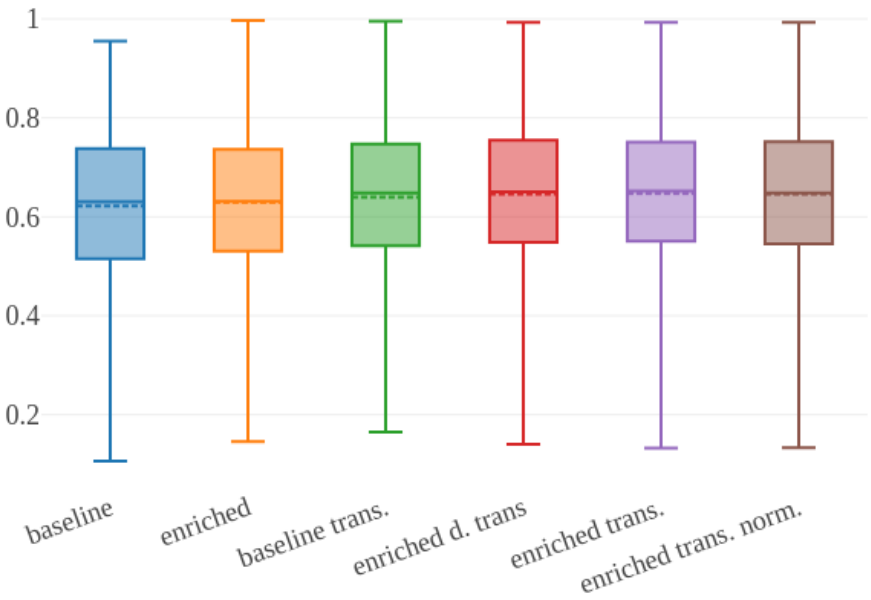


Figure 31: Transfer learning results compared to baseline scores.

6.3 Conclusion

We analysed the user behaviour of administrators, bots and normal users according to their message times within the Wikipedia social network. We investigated noticeable points of time between two continuous messages and the message rates across the three different roles. We observed a high number of low intervals and high message rates for bots. Surprisingly, administrators and normal users have intervals of 0 seconds. A more accurate evaluation within the millisecond scope could not be done, because the dataset only contained message timestamps with the smallest unit of seconds. The question that remains is how these zero interval values were created. It is possible that this could be an automatic response function provided by Wikipedia that is logged as a message. It is also possible that users created bots and did not separate them from their original accounts [26], and these are in fact automated activities but are logged as user activity. The message rate feature is more decisive. Here, human users have a visibly lower rate compared to bots.

In a machine learning scenario we see improvements when using the new timestamp features, as shown in the single evaluation. Incorporating interval and message rates also resulted in improvements. The entropy rates of daily and weekly messages did not improve the classification process. The k-core of a user returned

the best performance from the group of "other" features. Another feature that was not used in the final feature set was activity time, which originated from the idea of account registration used in research on other social networks. Since we do not know any account profile information, we used the date of the first and last sent messages from a given user for the activity time feature. However, for future predictions this feature has no informative value whatsoever. In the transfer learning scenario the best performance was achieved with a power law transformation of degree, k-core, intervals and message rates. Overall, we could not improve the classification of bots with timestamp features. The main reason for this is the occurrence of message interval lengths of zero across all three roles. Message rates provide more information, but only tells us if we are dealing with a human or bot users.

In this research, the massive number of normal users and the rarity of bots and thus rarity of bot activity should be kept in mind. As a final conclusion we can state that our three research questions are answered and the enriched feature set offers no advantage than the feature set only consisting of structural features with data of Wiki-Talk to the year 2015 . When identifying bots within a social network, what types of automated users one is looking for should be considered. Are these automated users voluntary , functional robots from the network itself or allowed bots, or do they present a manipulating or spam concern for the network? It should also be noted that a programmer who is aware of detection methods could program non-noticeable behaviour in message timing and rates.

7 Future work

The work presented in this study can be improved in various ways, but first an update of the dataset should be done. In the activity graphs we observed a rise in bot activities within the most recent years of the dataset. If the higher usage of bots continues in the years 2016 and 2017, more information can be observed and the classifier can be better trained. Because we identified the problem of human users with 0 second message intervals, a further investigation of the Wiki-Talk functionality is required to determine how these intervals between two messages can occur.

Additionally, a multi-label classification could be used to classify and label users into more than three classes. Here, a distribution of users with 0 second message intervals could be helpful to determine the extent of this behaviour. Another approach to classifying users could be observing message and response patterns. One potential new research question to explore is if there are any pattern observable for messages with 0 second intervals. Through multi-class classification, the problem of confusing identification between roles could be analysed within a confusion matrix to observe where the incorrect predictions are placed.

References

[1] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia, "Who is tweeting on twitter: Human, bot, or cyborg?" in *Proceedings of the 26th Annual Computer Security Applications Conference*, ser. ACSAC '10. New York, NY, USA: ACM, 2010, pp. 21–30. [Online]. Available: http://doi.acm.org/10.1145/1920261.1920265

[2] R. Heatherly and M. Kantarcioglu, "Extending the classification of nodes in social networks," in *Proceedings of 2011 IEEE International Conference on Intelligence and Security Informatics*, July 2011, pp. 77–82.

[3] C. Buntain and J. Golbeck, "Identifying social roles in reddit using network structure," in *Proceedings of the 23rd International Conference on World Wide Web*, ser. WWW '14 Companion. New York, NY, USA: ACM, 2014, pp. 615–620. [Online]. Available: http //doi.acm.org/10.1145/2567948.2579231

[4] G. Bai, L. Liu, B. Sun, and J. Fang, "A survey of user classification in social networks," in *2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, Sept 2015, pp. 1038–1041.

[5] J. Sun, J. Kunegis, and S. Staab, "Predicting user roles in social networks using transfer learning with feature transformation," *CoRR*, vol. abs/1611.02941, 2016. [Online]. Available: http://arxiv.org/abs/1611.02941

[6] Wikipedia. (2018) User access levels - wikipedia, the free encyclopedia. [Online]. Available: https //en.wikipedia.org/wiki/Wikipedia:User_access_levels

[7] ——. (2018) Citation bot - wikipedia, the free encyclopedia. [Online]. Available: https://en.wikipedia.org/wiki/User Citation_bot

[8] J. Sun and J. Kunegis. (2016) Wiki-talk datasets. [Online]. Available: https://zenodo.org/record/49561#.Wz3o6PZuJNM

[9] B. Bollobás, *Graph Theory - An Introductory Course*. Springer-Verlag New York, 1979.

[10] A. Leontjeva, M. Goldszmidt, Y. Xie, F. Yu, and M. Abadi, "Early security classification of skype users via machine learning," in *Proceedings of the 2013 ACM Workshop on Artificial Intelligence and Security*, ser. AISec '13. New York, NY, USA: ACM, 2013, pp. 35–44. [Online]. Available: http://doi.acm.org/10.1145/2517312.2517322

[11] S. H. F. L. L. M. H. S. M. Kitsak, L. Gallos and H. Makse, "Identification of influential spreaders in complex networks," *J. ACM, vol.30, no.3, pp 417-427*, 1983.

[12] F. D. Malliaros, A. N. Papadopoulos, and M. Vazirgiannis, "Core decomposition in graphs: Concepts, algorithms and applications."

[13] M. P. O'Brien and B. D. Sullivan, "Locally estimating core numbers," *IEEE International Conference on Data Mining*, pp. 460–469, 2014.

[14] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, Oct 2010.

[15] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861 – 874, 2006, rOC Analysis in Pattern Recognition. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016786550500303X

[16] H. Welser, E. Gleave, D. Fisher, and M. Smith, "Visualizing the signatures of social roles in online discussion groups," vol. 8, pp. 1–31, 01 2007.

[17] E. Ferrara, O. Varol, C. Davis, F. Menczer, and A. Flammini, "The rise of social bots," *Commun. ACM*, vol. 59, no. 7, pp. 96–104, Jun. 2016. [Online]. Available: http://doi.acm.org/10.1145/2818717

[18] J. H. Wetstone and S. R. Nayyar, "I spot a bot: Building a binary classifier to detect bots on twitter," *CS 229 Final Project Report*, 2017. [Online]. Available: http://cs229.stanford.edu/proj2017/final-reports/5240610.pdf

[19] I. Pozzana and E. Ferrara, "Measuring bot and human behavioral dynamics," *CoRR*, vol. abs/1802.04286, 2018. [Online]. Available: http://arxiv.org/abs/1802.04286

[20] S. Gianvecchio, M. Xie, Z. Wu, and H. Wang, "Humans and bots in internet chat: Measurement, analysis, and automated classification," *IEEE/ACM Transactions on Networking*, vol. 19, no. 5, pp. 1557–1571, Oct 2011.

[21] S. E. Christian Hadiwijaya Saputra, Erwin Adi, "Comparison of classification algorithms to tell bots and humans apart."

[22] A. Clauset, C. R. Shalizi, and M. E. J. Newman, "Power-law distributions in empirical data," *SIAM Rev.*, vol. 51, no. 4, pp. 661–703, Nov. 2009. [Online]. Available: http://dx.doi.org/10.1137/070710111

[23] J. Sun. Tranet. [Online]. Available: https://github.com/yfiua/TraNet

[24] plotly - data visualization tool. [Online]. Available: https://plot.ly/

[25] scikit-learn - the machine learning library. [Online]. Available: http://scikit-learn.org/stable/index.html

[26] Wikipedia. (2018) Bot policy - wikipedia. [Online]. Available: https://en.wikipedia.org/wiki/Wikipedia:Bot_policy

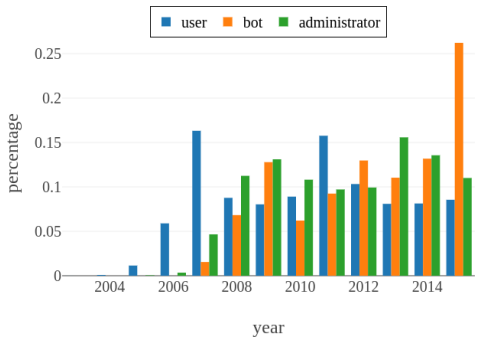
List of Tables

List of Figures

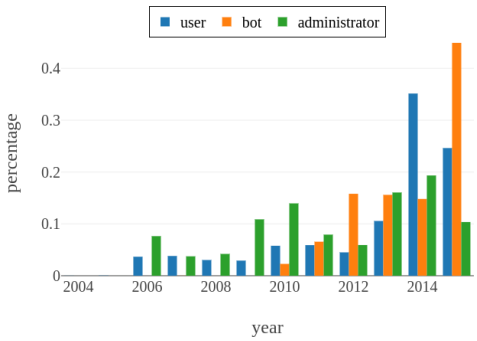
Appendix

Dataset observations

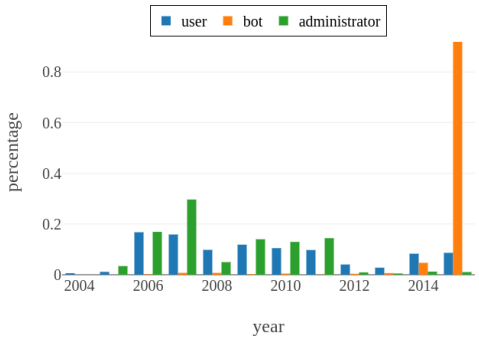
Role activity



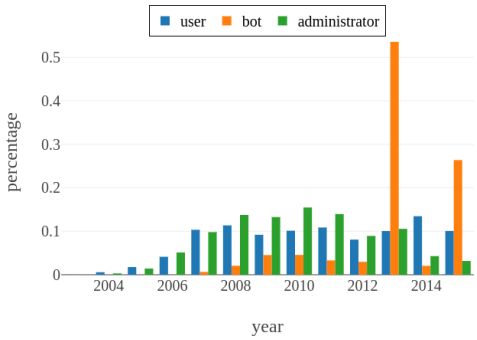
(a) ar



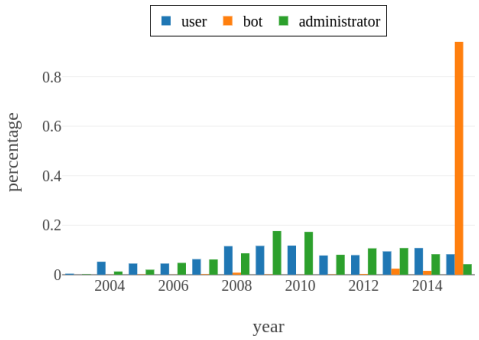
(b) bn



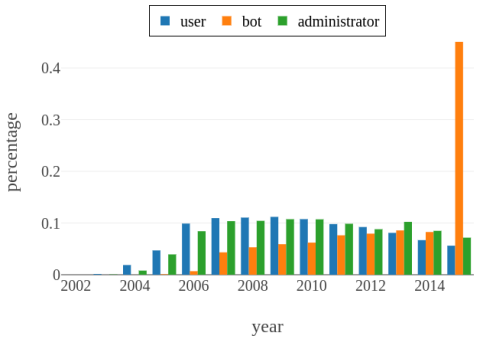
(c) br



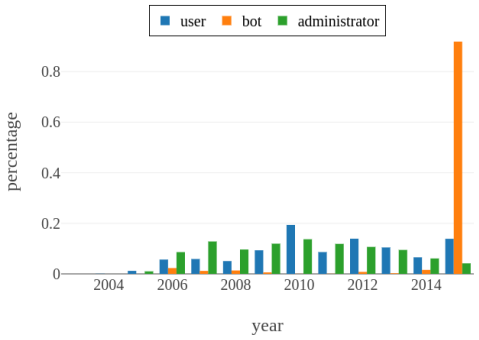
(d) ca



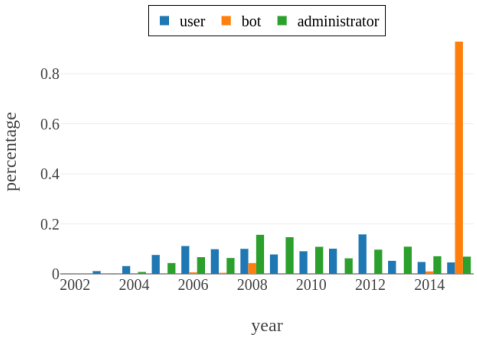
(e) cy



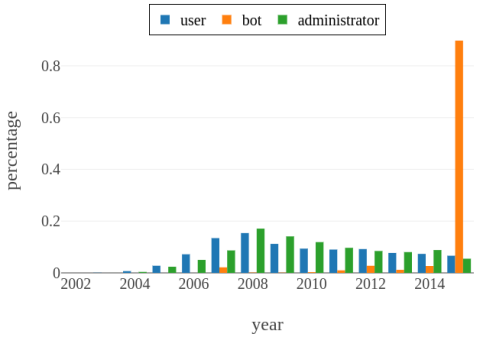
(f) de



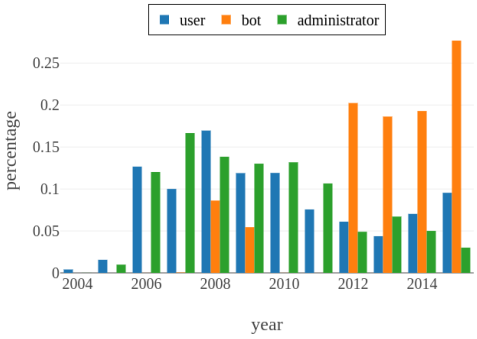
(g) el



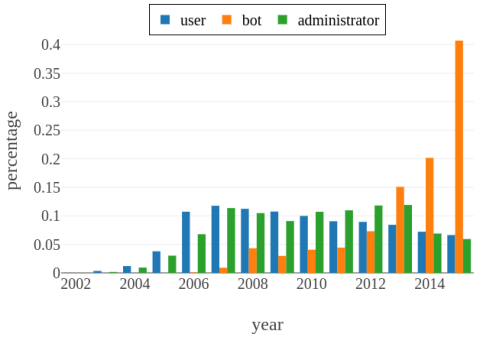
(h) eo



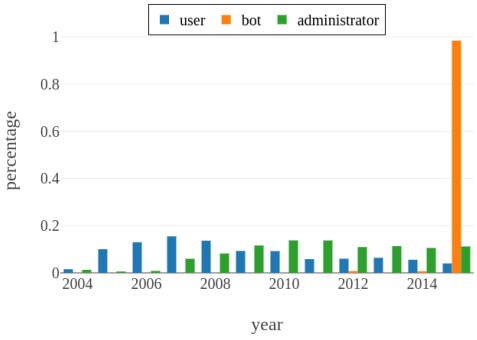
(i) es



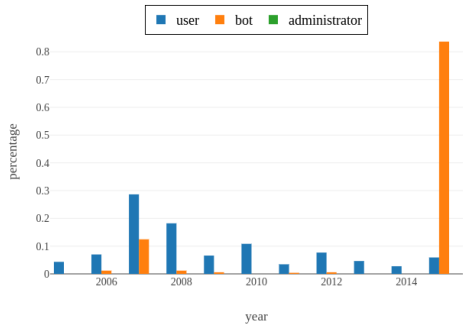
(j) eu



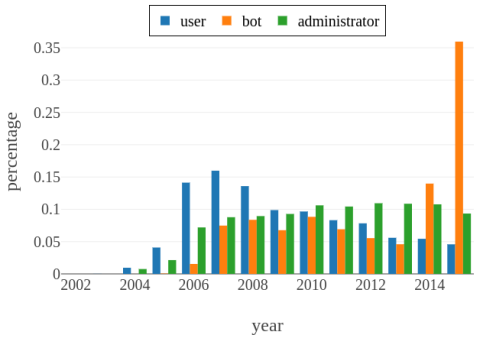
(k) fr



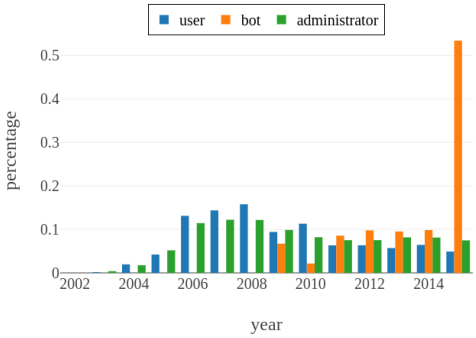
(l) gl



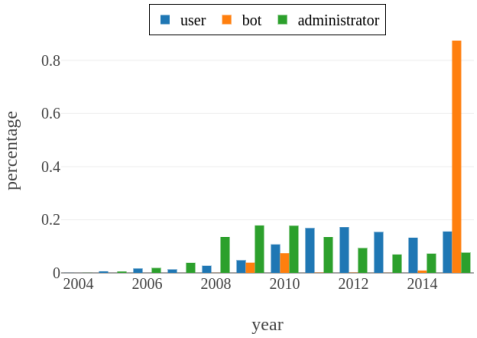
(m) ht



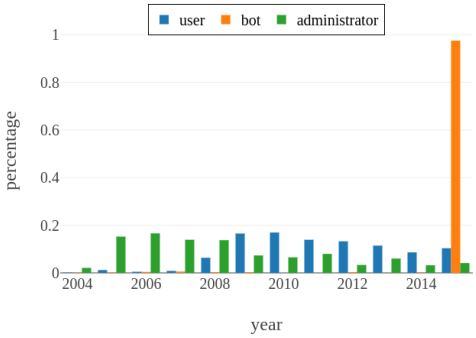
(n) it



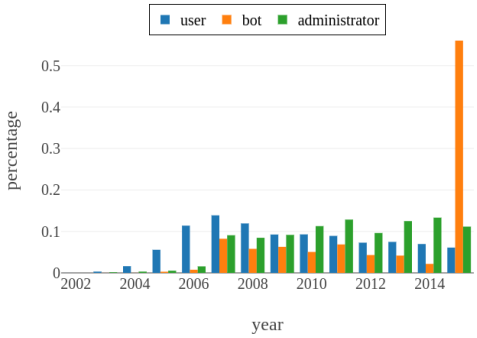
(o) ja



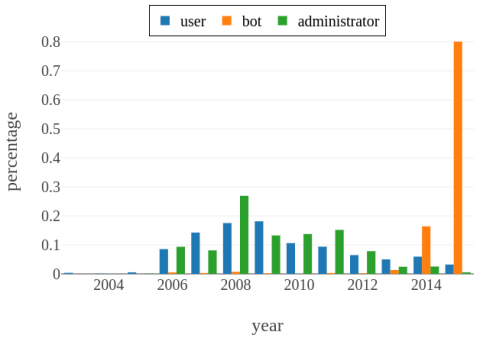
(p) lv



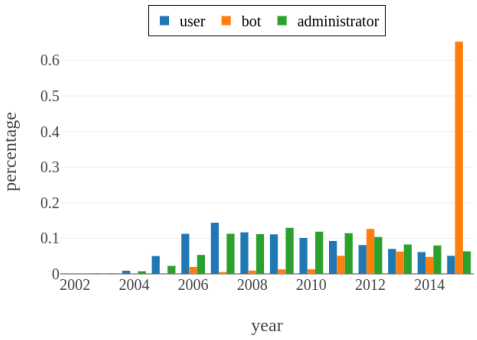
(q) nds



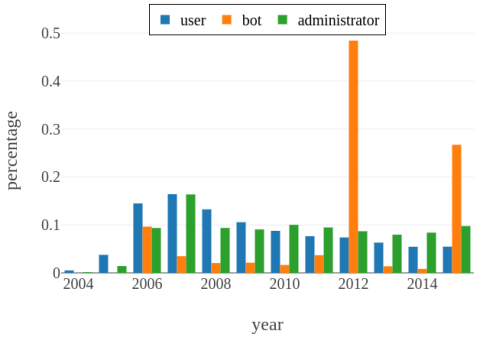
(r) nl



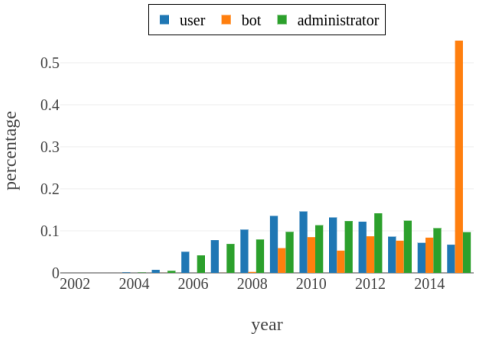
(s) oc



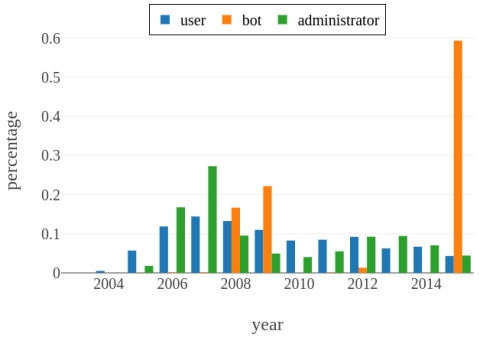
(t) pl



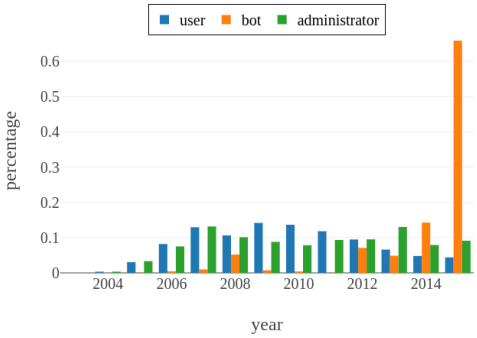
(u) pt



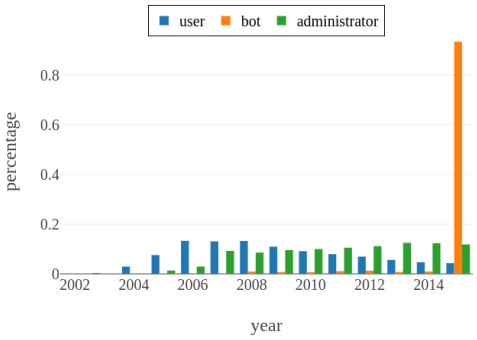
(v) ru



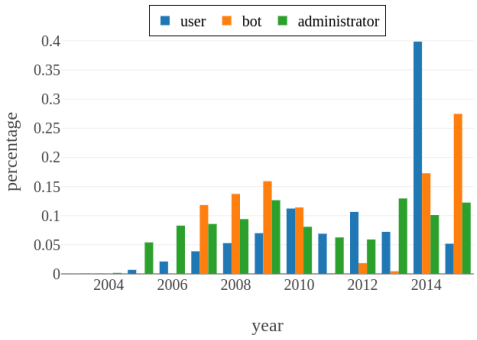
(w) sk



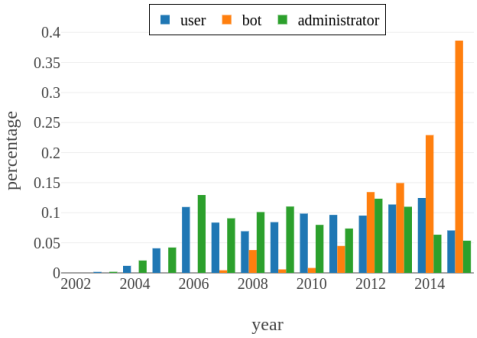
(x) sr



(y) sv



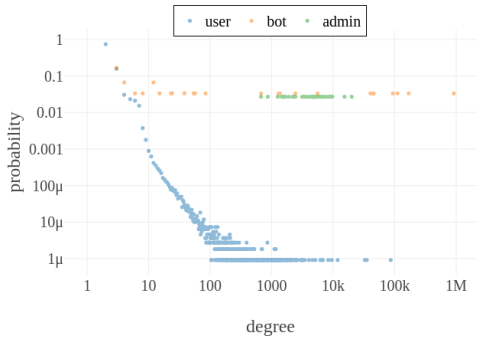
(z) vi



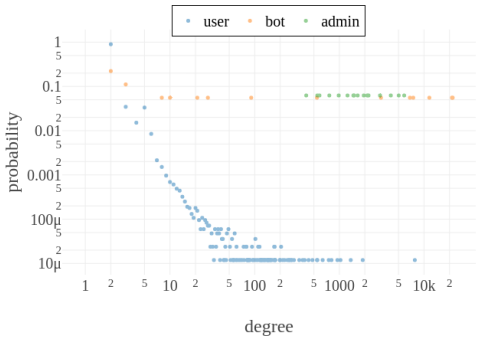
(aa) zh

Figure 32: Activity by role.

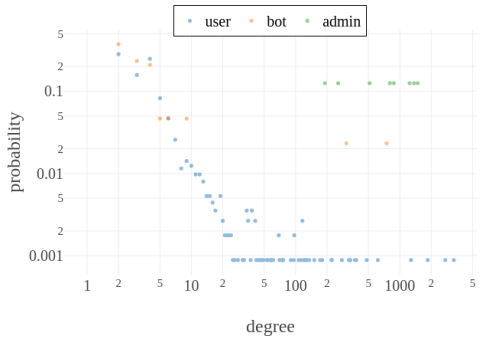
Degree distribution



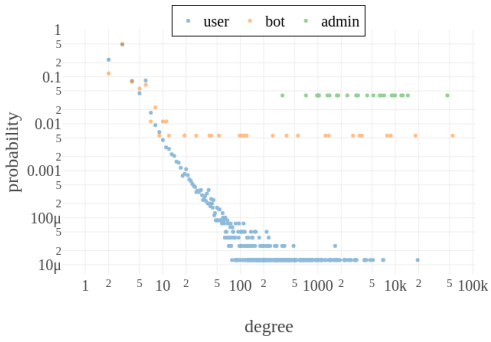
(a) ar



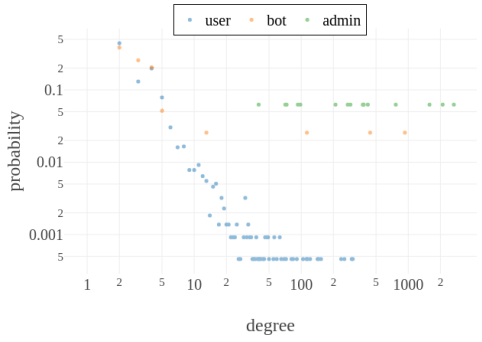
(b) bn



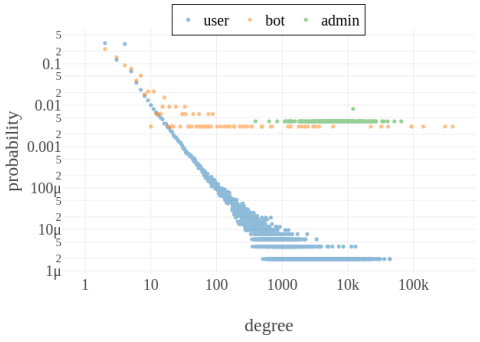
(c) br



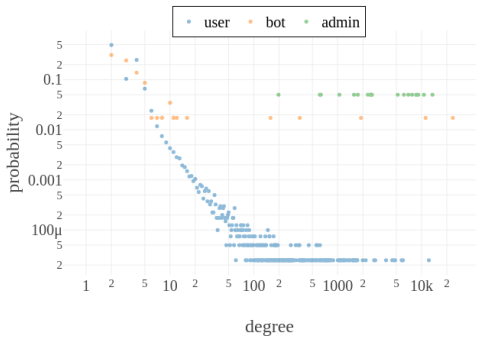
(d) ca



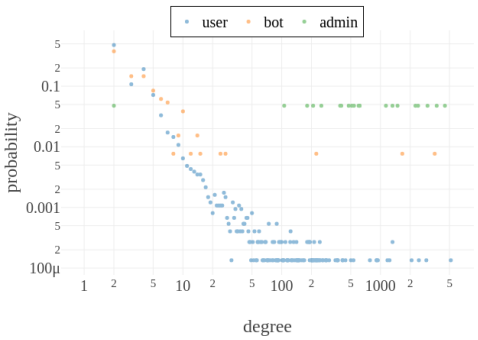
(e) cy



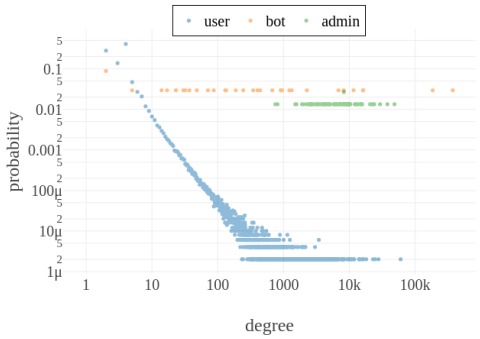
(f) de



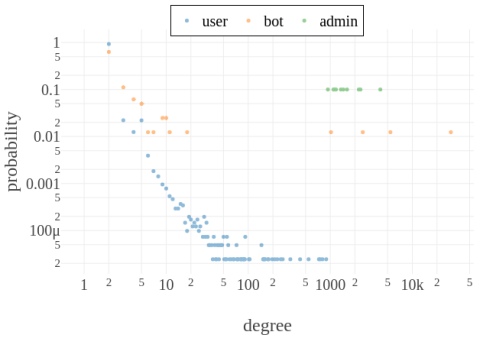
(g) el



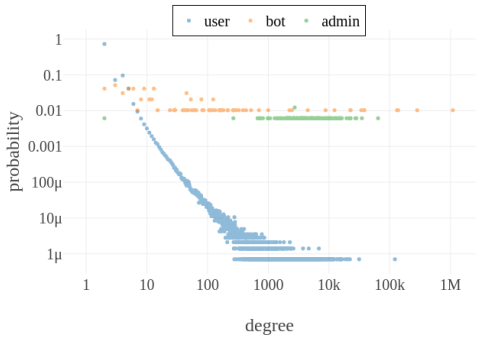
(h) eo



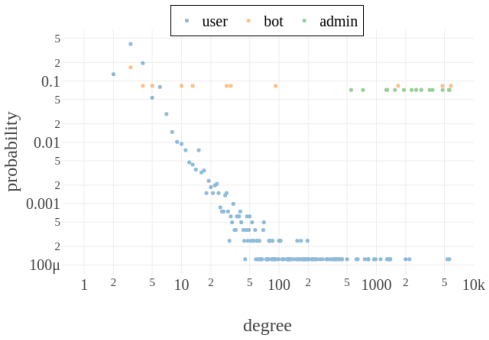
(i) es



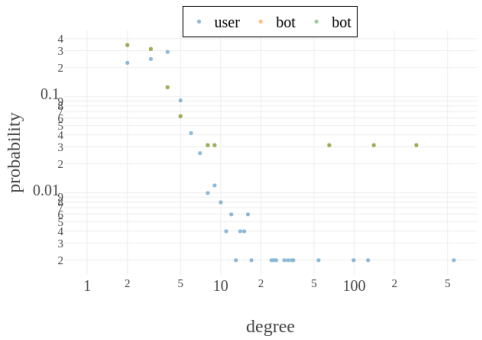
(j) eu



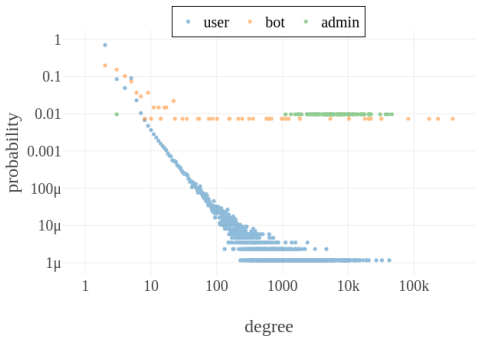
(k) fr



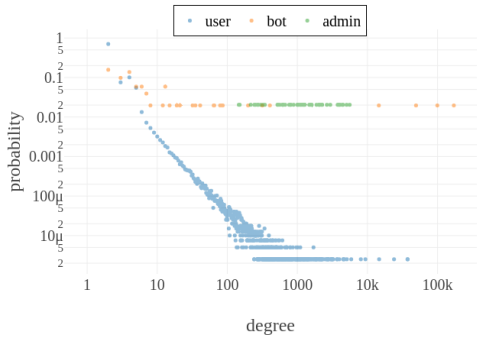
(l) gl



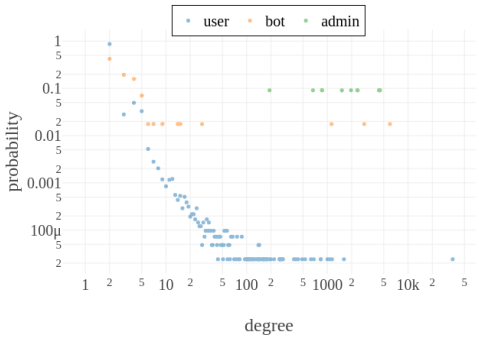
(m) ht



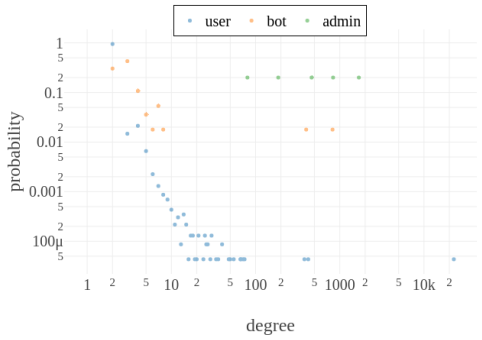
(n) it



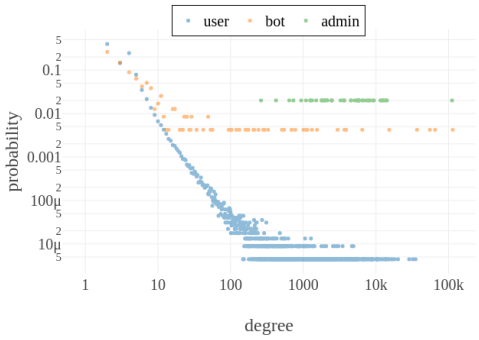
(o) ja



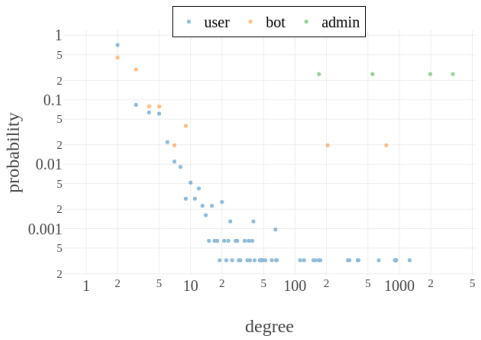
(p) lv



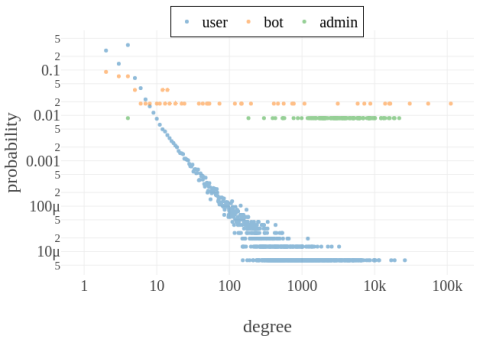
(q) nds



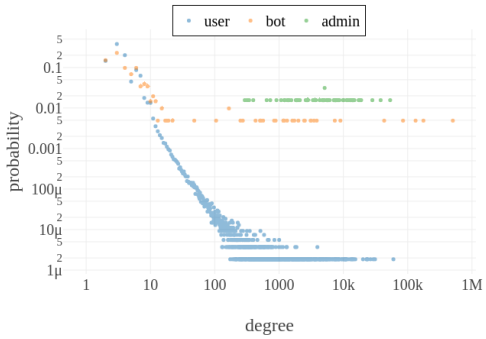
(r) nl



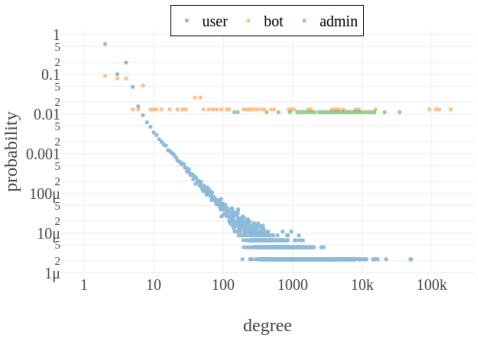
(s) oc



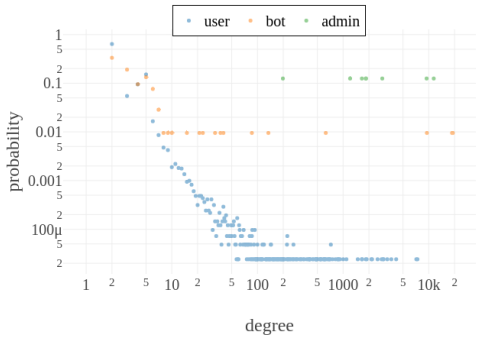
(t) pl



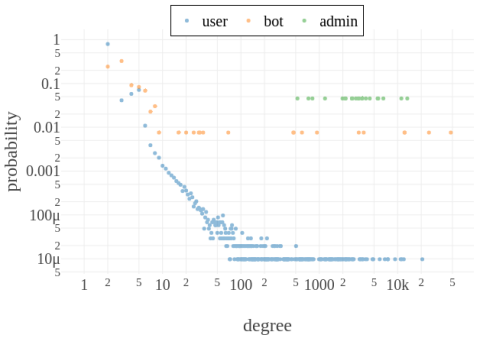
(u) pt



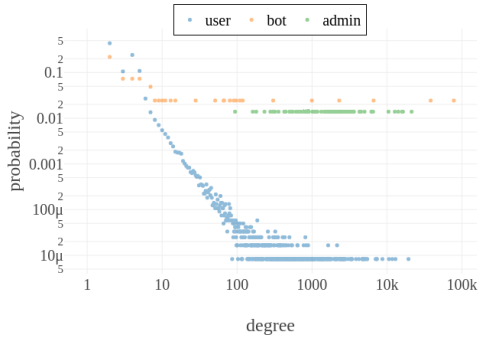
(v) ru



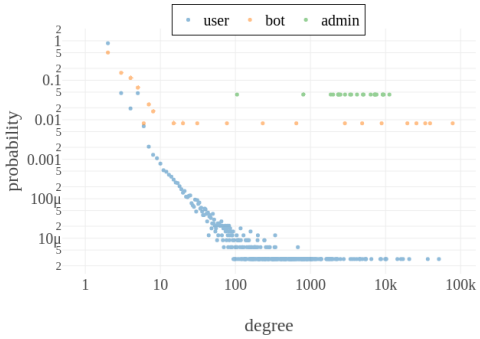
(w) sk



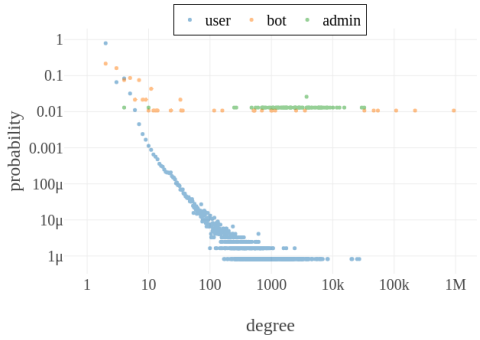
(x) sr



(y) sv



(z) vi



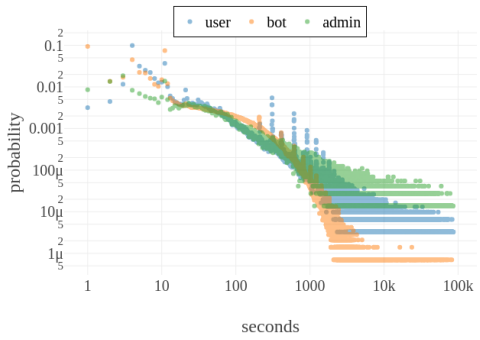
(aa) zh

Figure 33: Degree distribution by role.

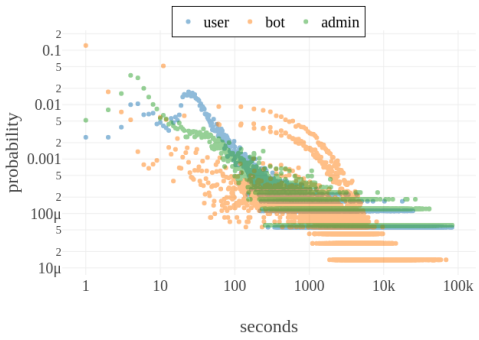
Language	User	Bot	Admin
ar	2.02	47,706.63	4,955.51
bn	1.65	4,042.83	2,082.0
br	17.38	26.74	840.25
ca	5.47	583.39	6,542.52
cy	4.65	39.97	611.81
de	19.49	3,549.89	8,884.72
el	6.18	644.19	4,737.65
en	12.82	10,843.95	6,604.0
eo	8.55	45.11	1,175.81
es	8.32	18,473.82	8,526.6
eu	1.48	476.1	1,734.2
fr	4.51	19,011.47	6,384.0
gl	9.52	1036.75	2,737.57
ht	4.96	17.5	
it	4.89	7,505.37	8,528.0
ja	4.15	6,581.1	1,590.0
lv	2.75	177.47	2,187.0
nds	2.18	23.89	648.0
nl	10.79	1,333.14	7,227.0
oc	4.89	20.71	1,486.0
pl	12.19	4,970.07	4,755.0
pt	6.22	4,846.6	7,637.0
ru	7.55	7,629.71	5,857.0
sk	4.47	462.05	3,807.0
sr	4.22	811.67	3,826.0
sv	7.08	3,103.22	2,982.0
vi	2.62	1,742.8	4,910.0
zh	2.33	15,155.01	4,087.0

Table 8: Mean degree distribution by role over all datasets.

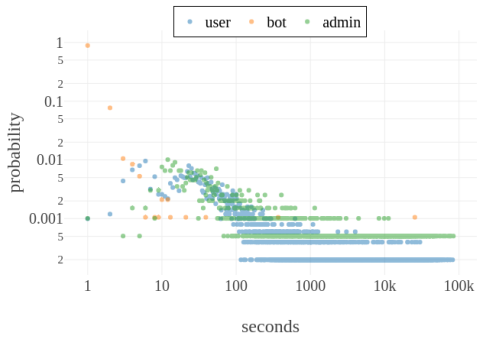
Interval distribtion



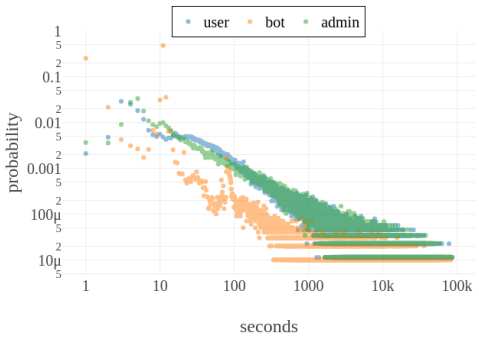
(a) ar



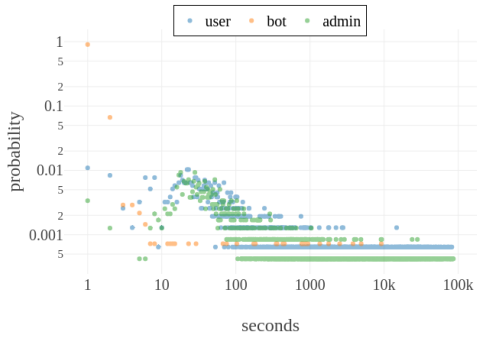
(b) bn



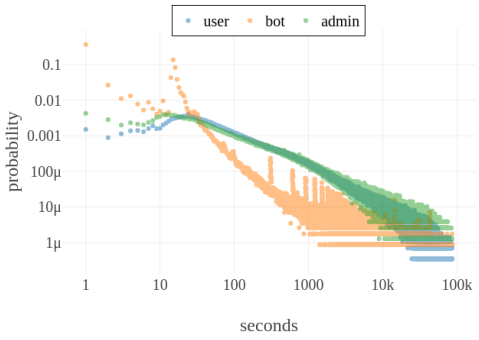
(c) br



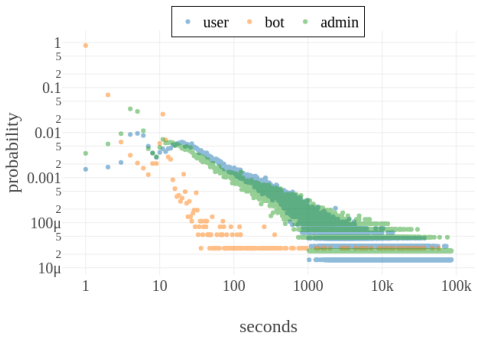
(d) ca



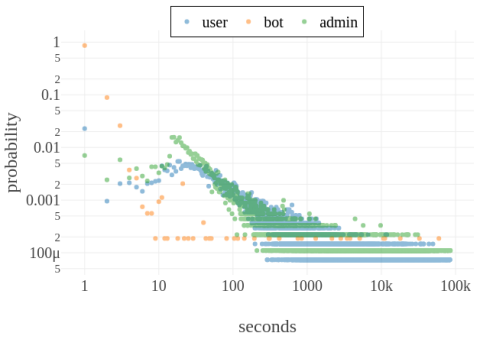
(e) cy



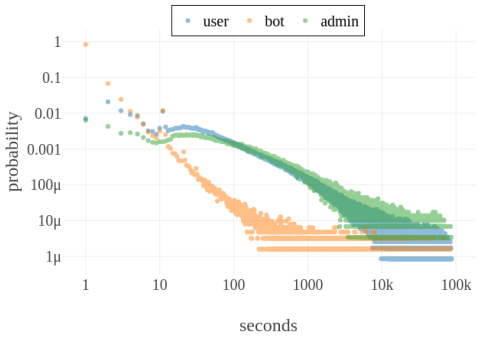
(f) de



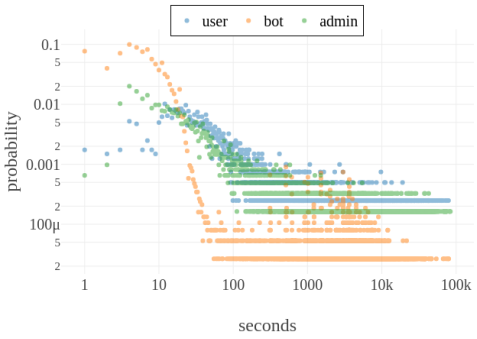
(g) el



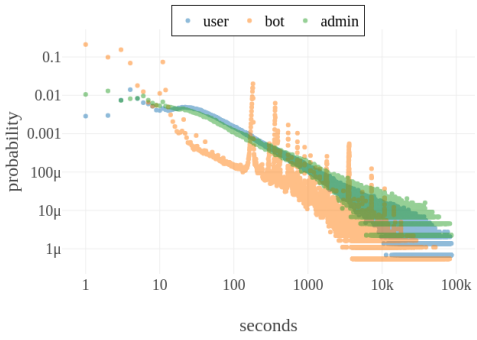
(h) eo



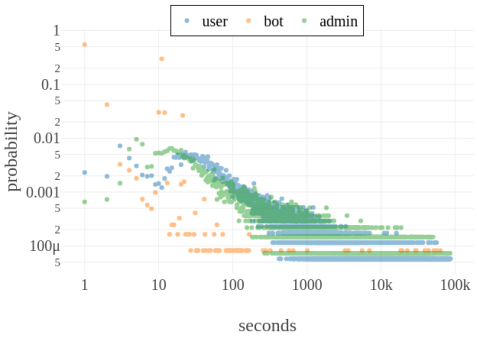
(i) es



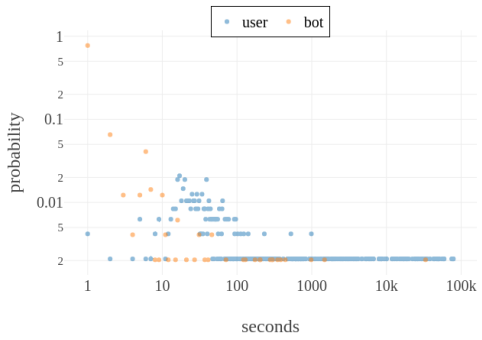
(j) eu



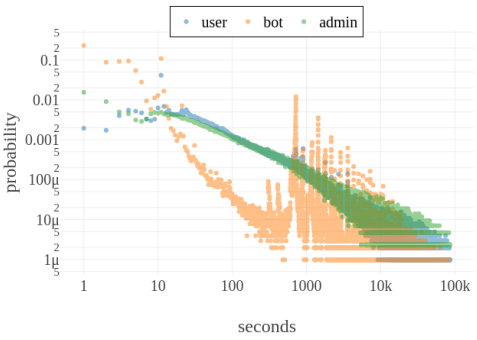
(k) fr



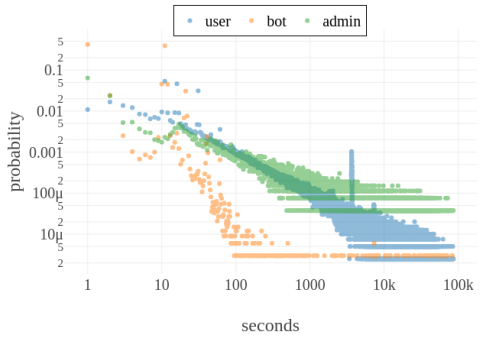
(l) gl



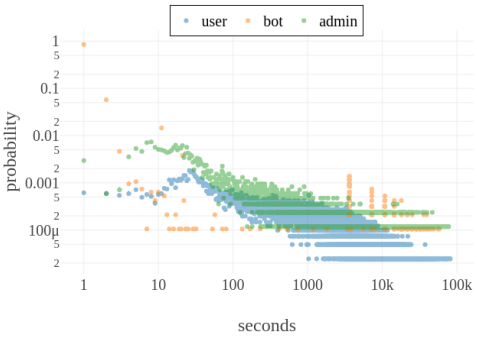
(m) ht



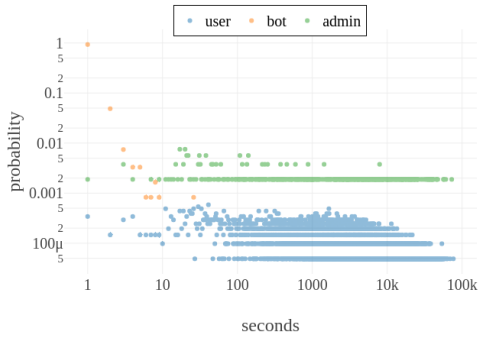
(n) it



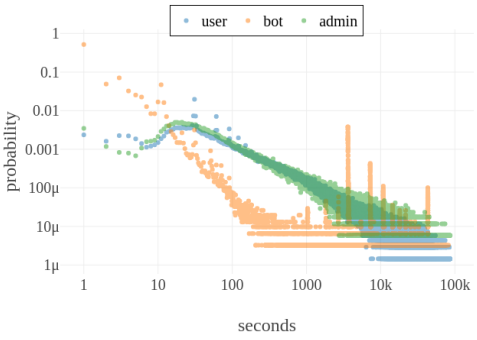
(o) ja



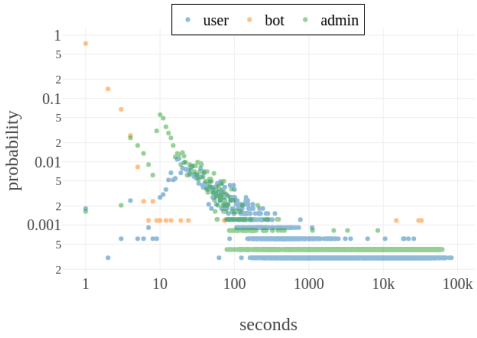
(p) lv



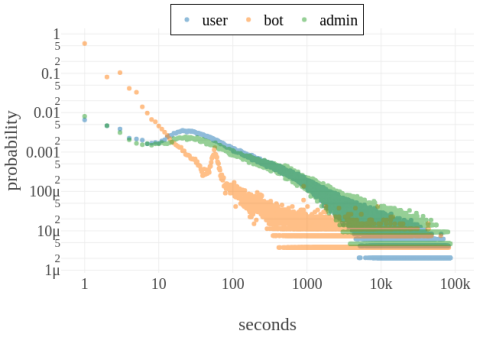
(q) nds



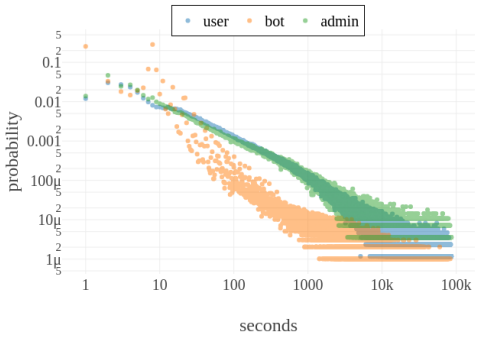
(r) nl



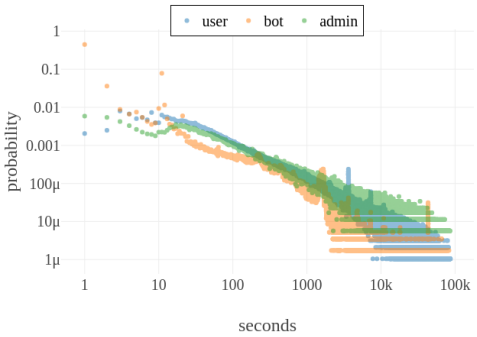
(s) oc



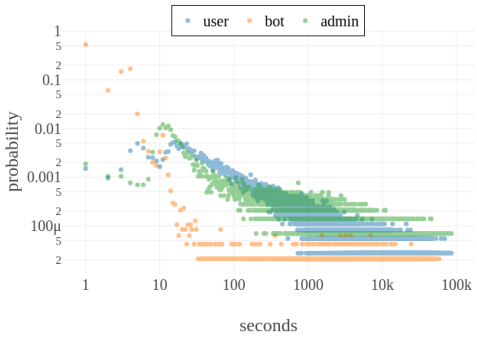
(t) pl



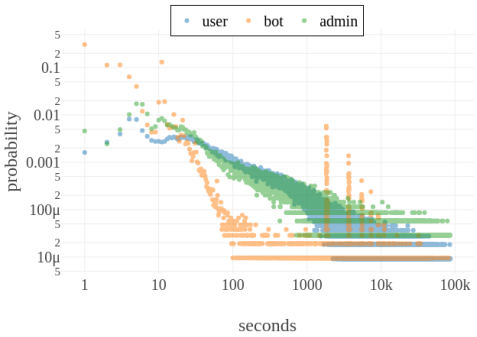
(u) pt



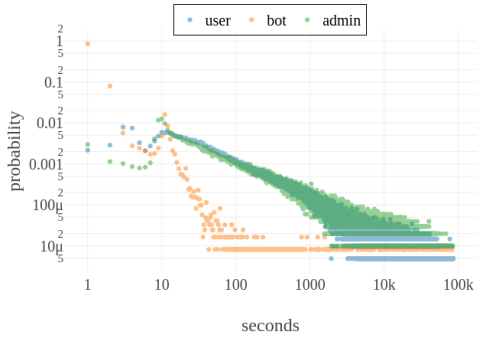
(v) ru



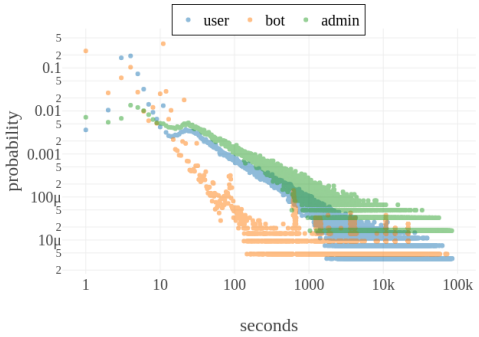
(w) sk



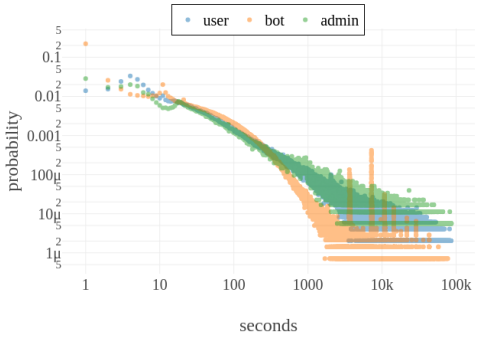
(x) sr



(y) sv



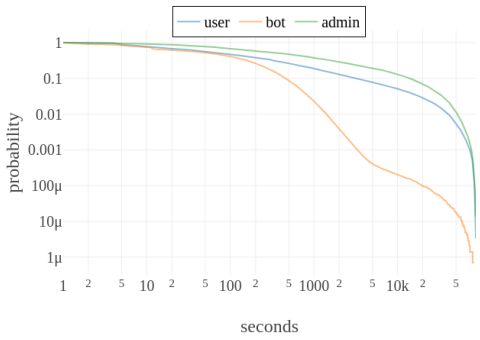
(z) vi



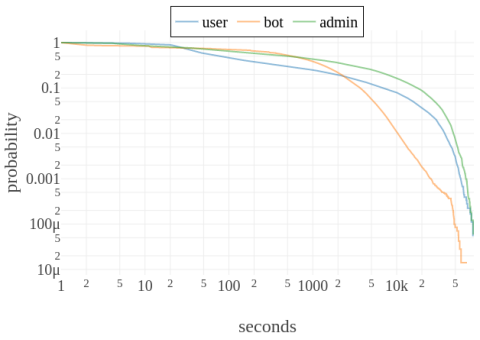
(aa) zh

Figure 34: Interval distribution by role.

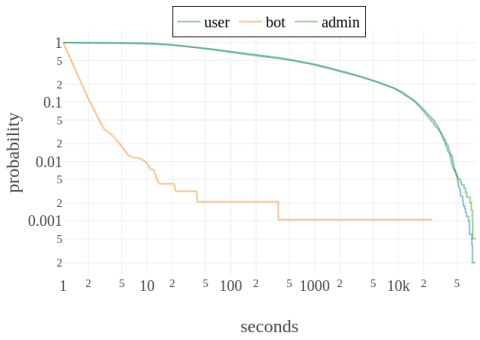
Interval CCDF



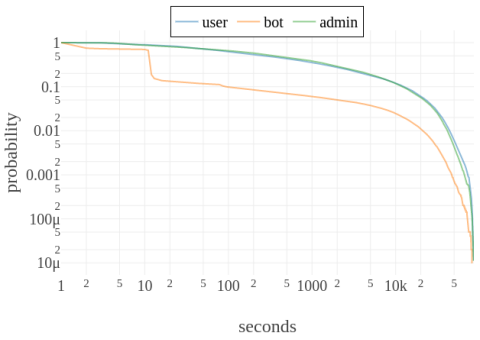
(a) ar



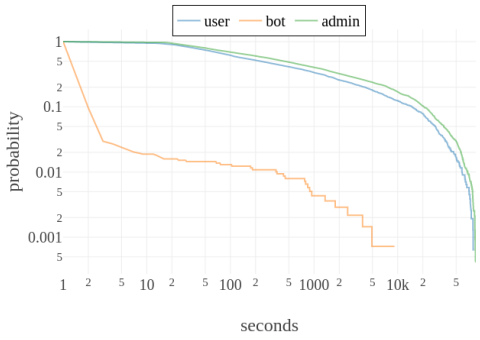
(b) bn



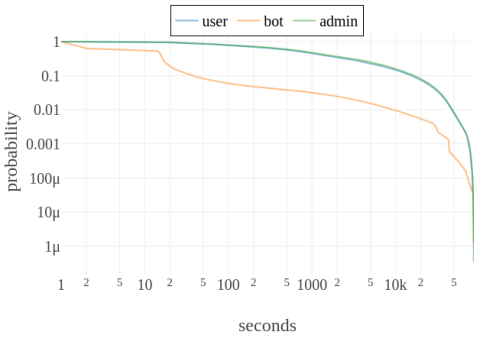
(c) br



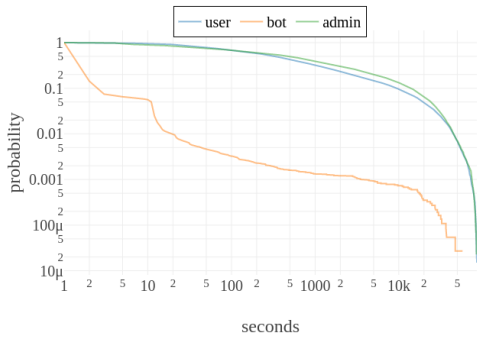
(d) ca



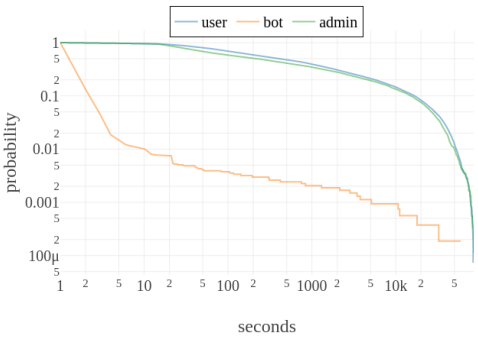
(e) cy



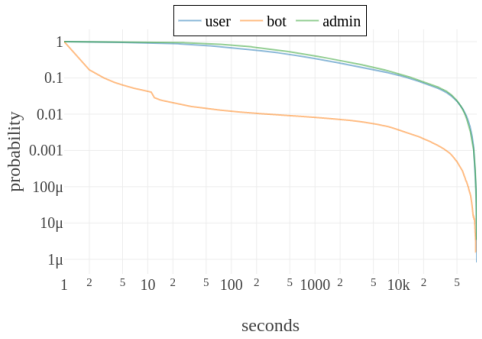
(f) de



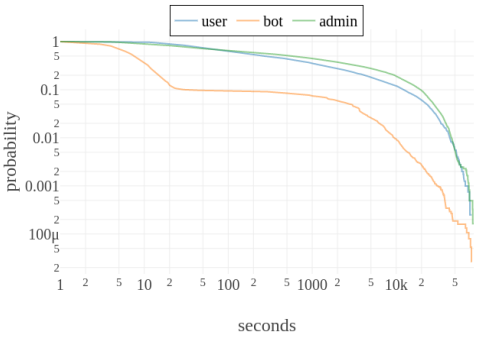
(g) el



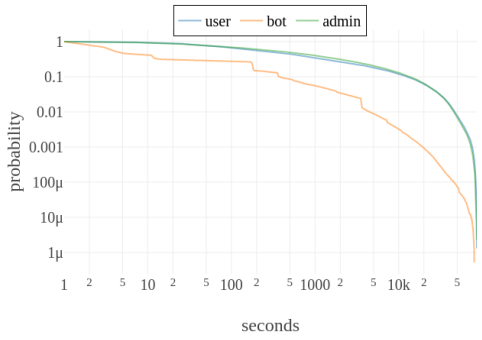
(h) eo



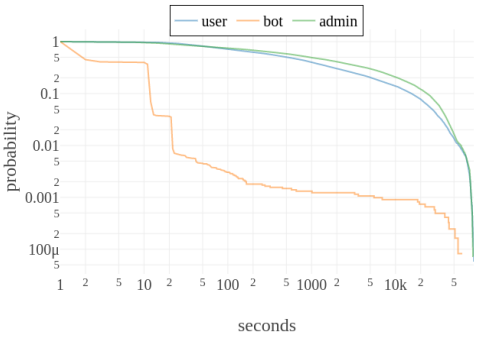
(i) es



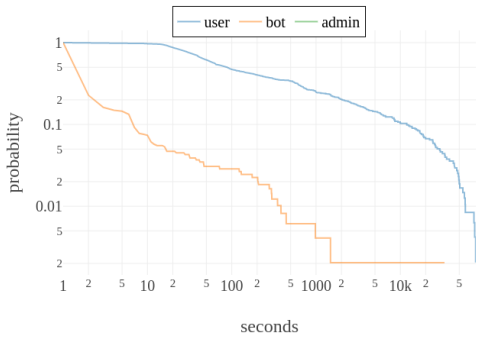
(j) eu



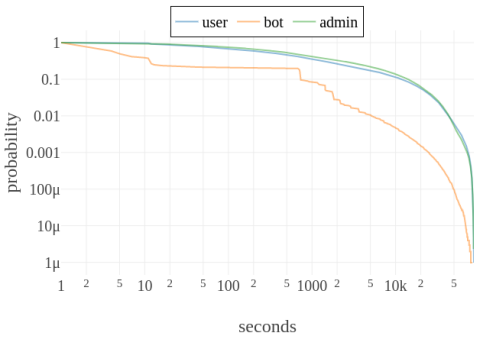
(k) fr



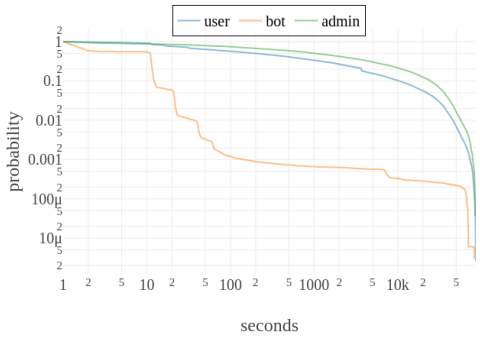
(l) gl



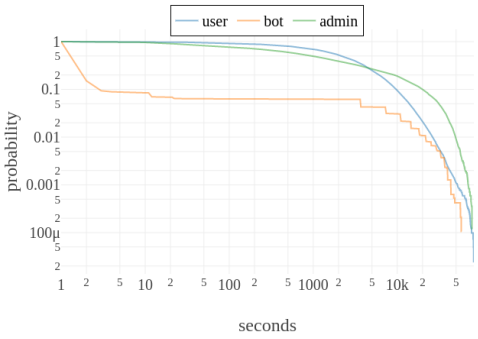
(m) ht



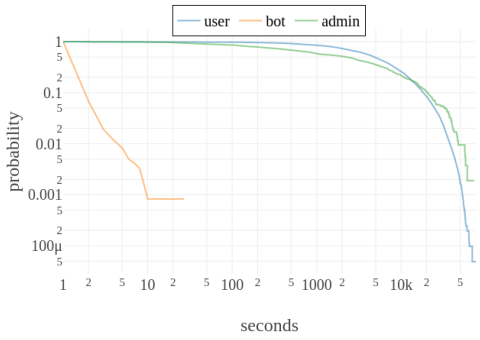
(n) it



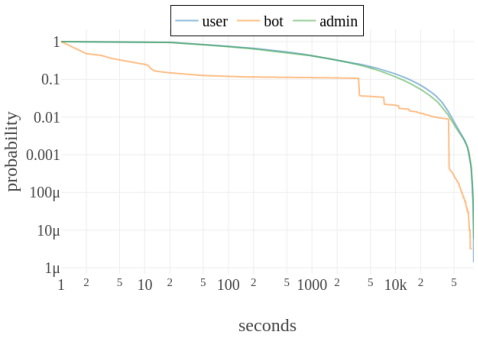
(o) ja



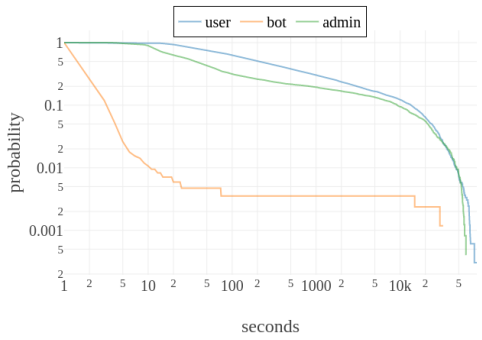
(p) lv



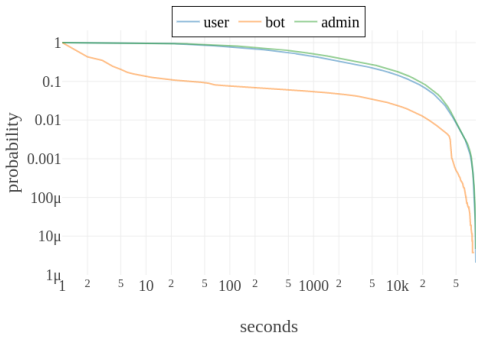
(q) nds



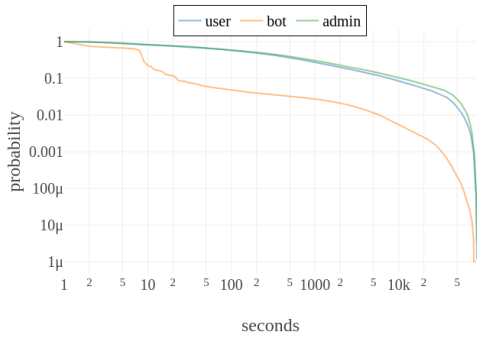
(r) nl



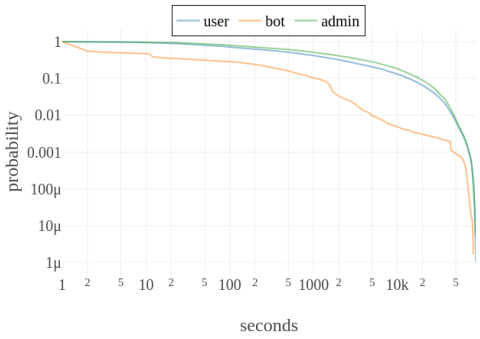
(s) oc



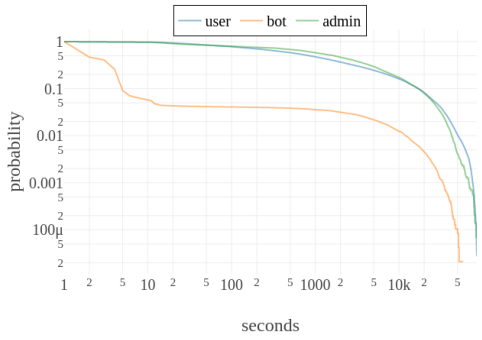
(t) pl



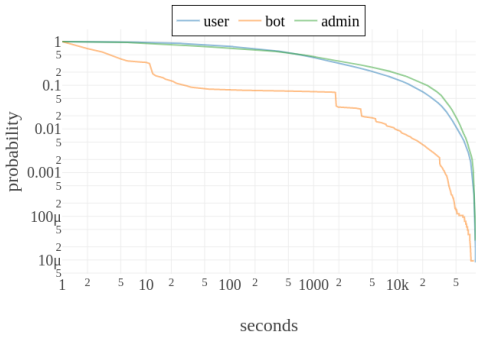
(u) pt



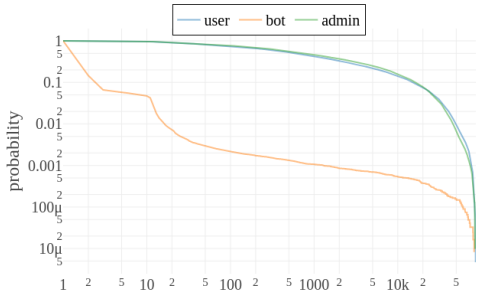
(v) ru



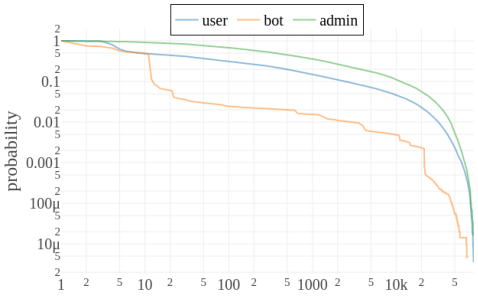
(w) sk



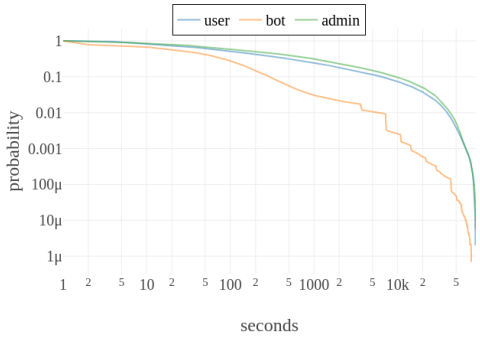
(x) sr



(y) sv



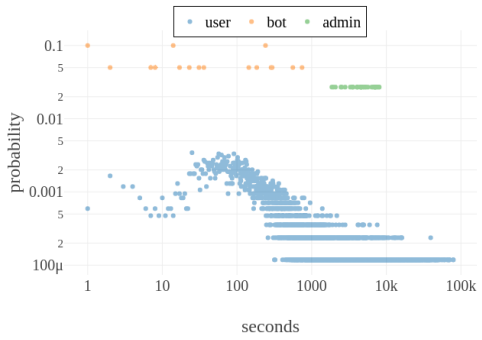
(z) vi



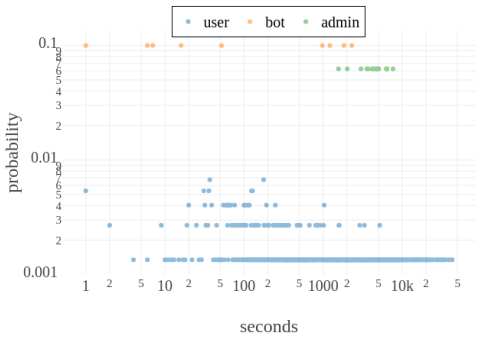
(aa) zh

Figure 35: Interval CCDF by role.

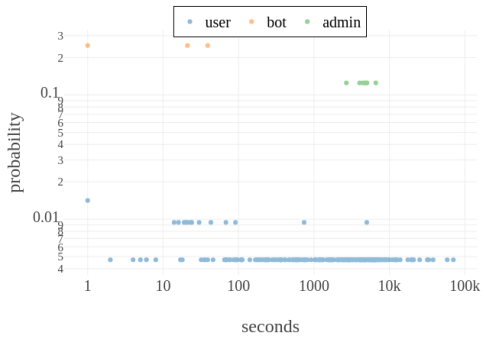
Arithmetic mean of user interval distribution



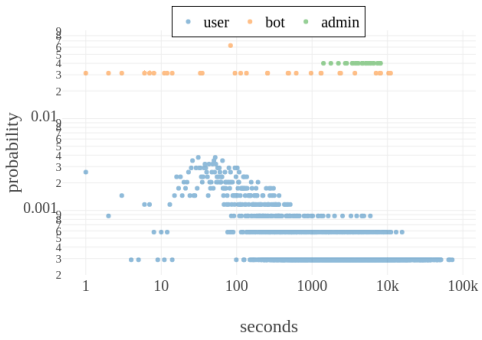
(a) ar



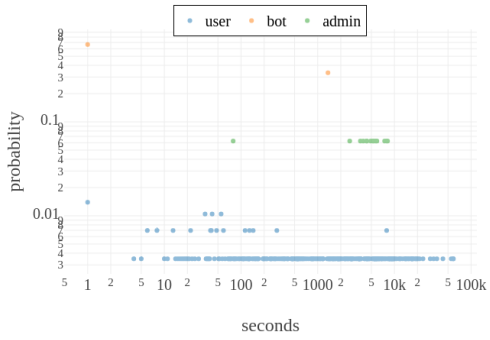
(b) bn



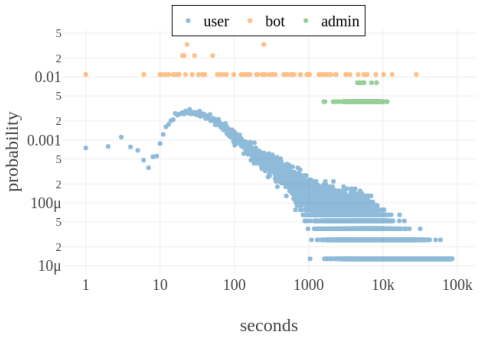
(c) br



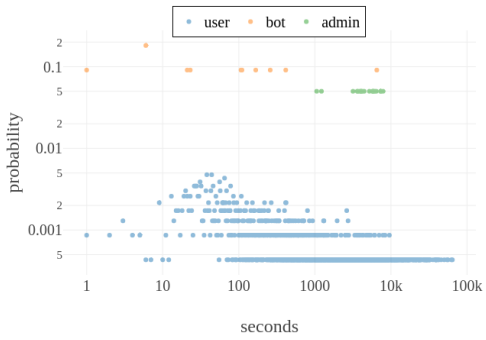
(d) ca



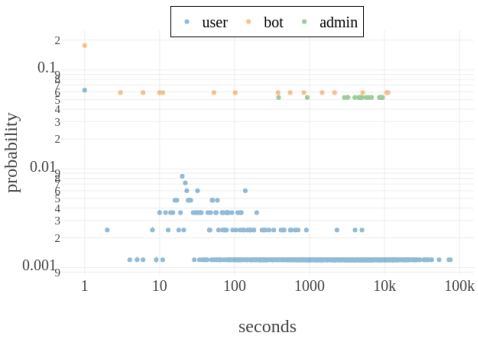
(e) cy



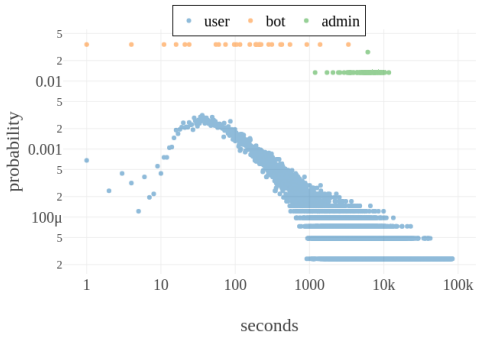
(f) de



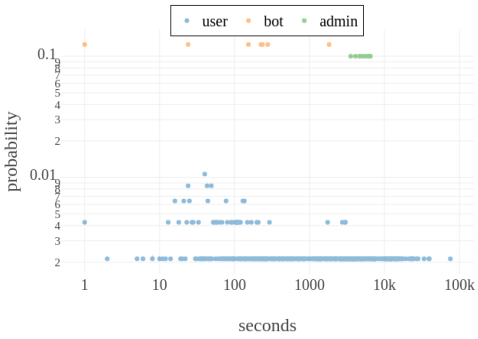
(g) el



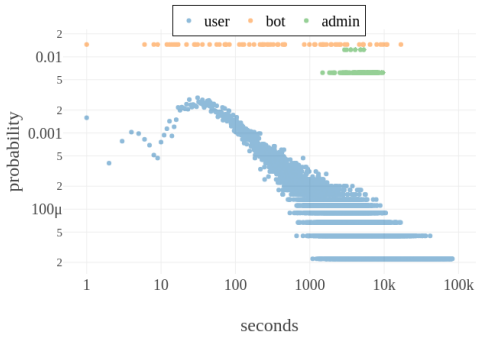
(h) eo



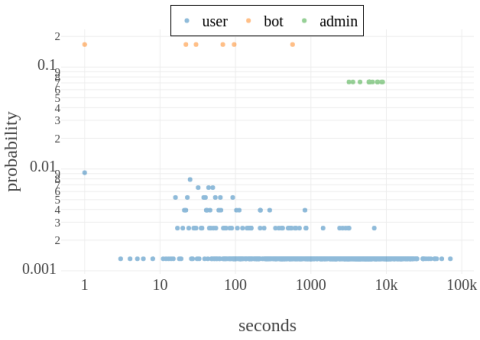
(i) es



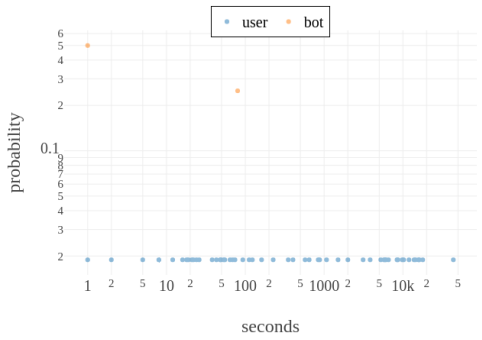
(j) eu



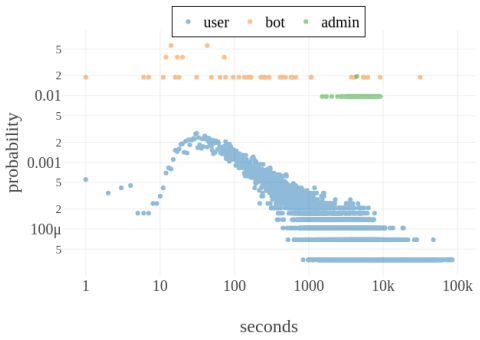
(k) fr



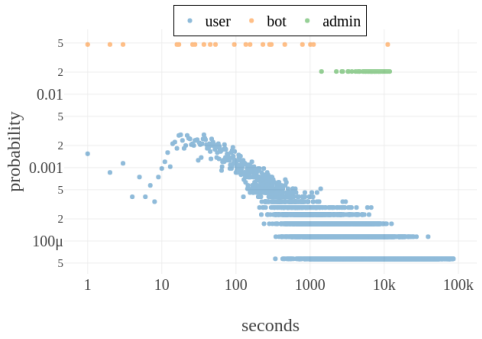
(l) gl



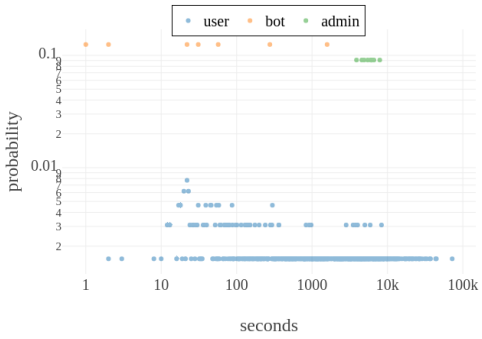
(m) ht



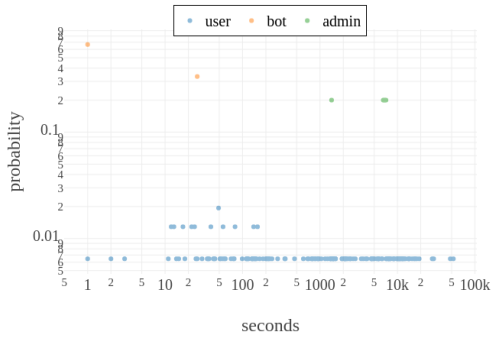
(n) it



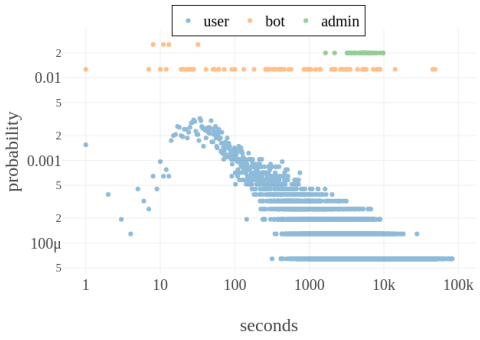
(o) ja



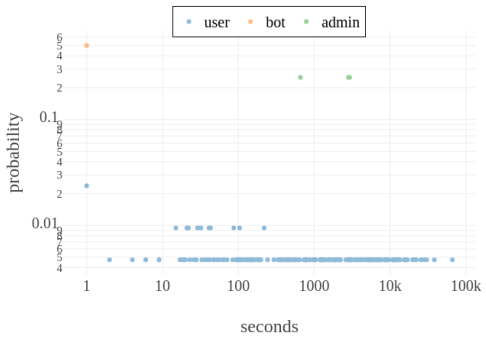
(p) lv



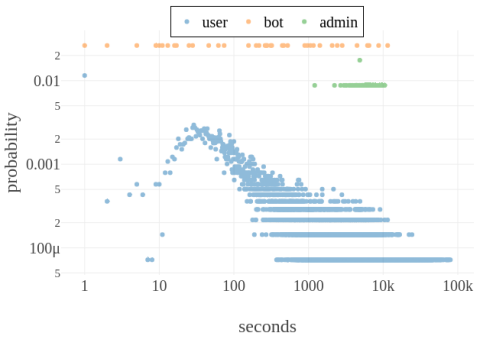
(q) nds



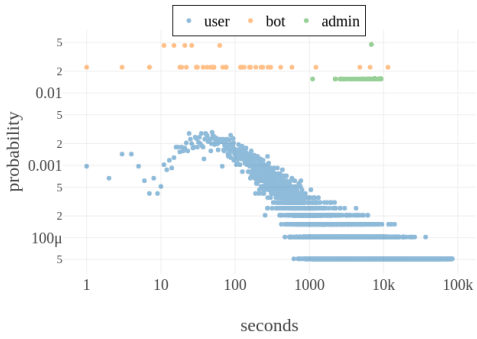
(r) nl



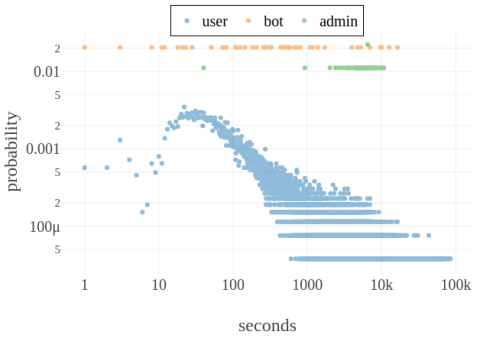
(s) oc



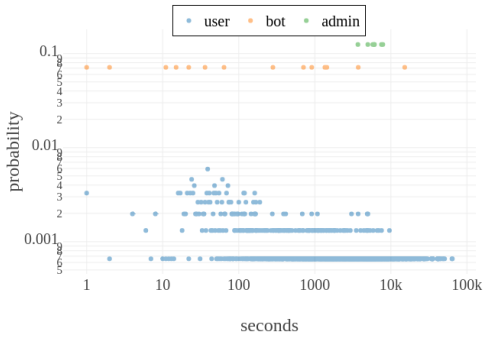
(t) pl



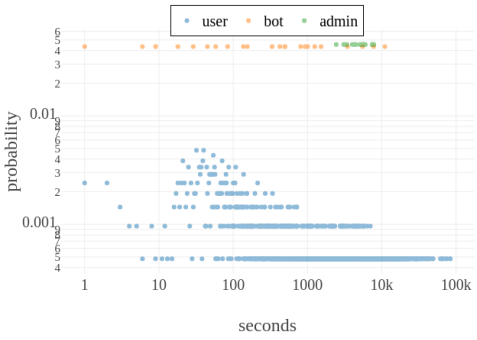
(u) pt



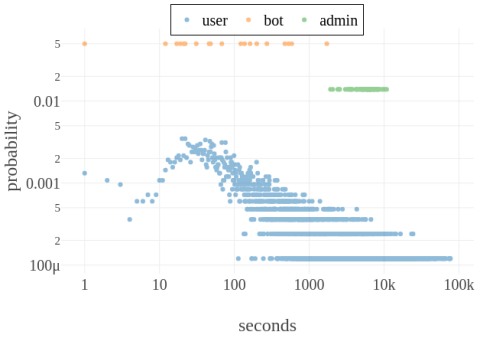
(v) ru



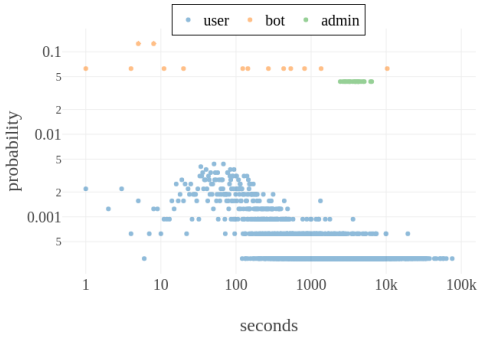
(w) sk



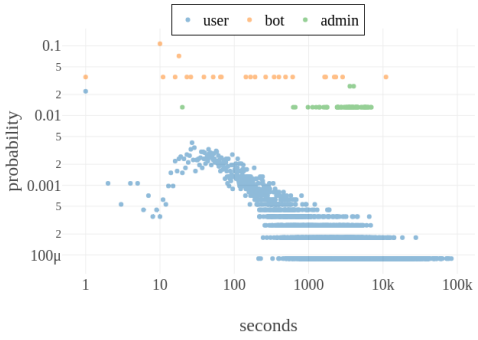
(x) sr



(y) sv



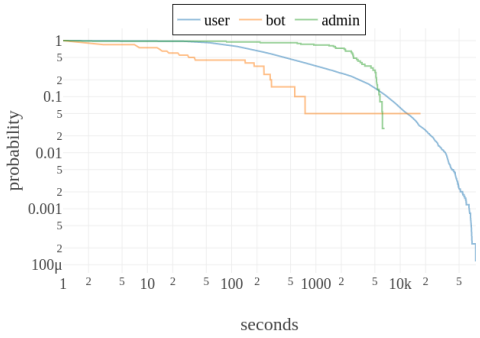
(z) vi



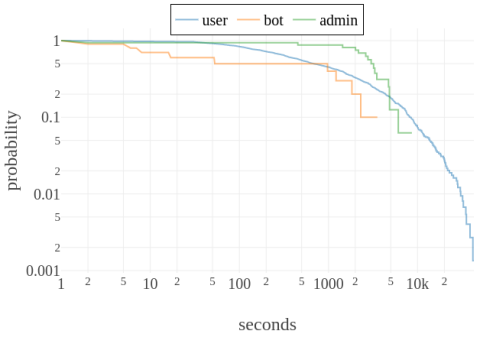
(aa) zh

Figure 36: Interval distribution by role.

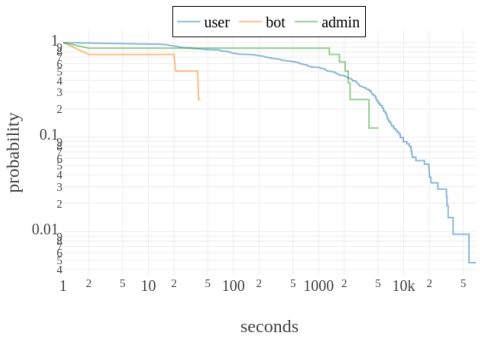
Arithmetic mean of user interval CCDF



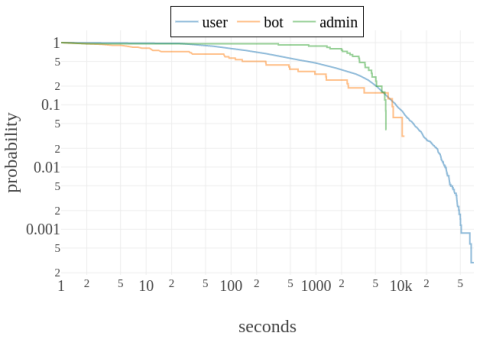
(a) ar



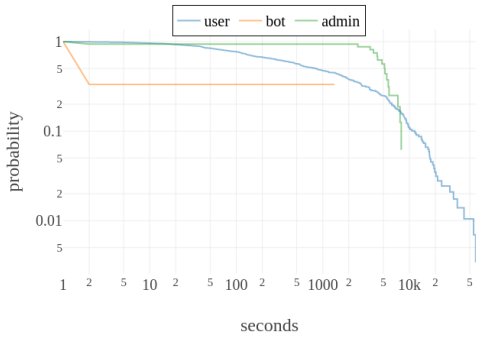
(b) bn



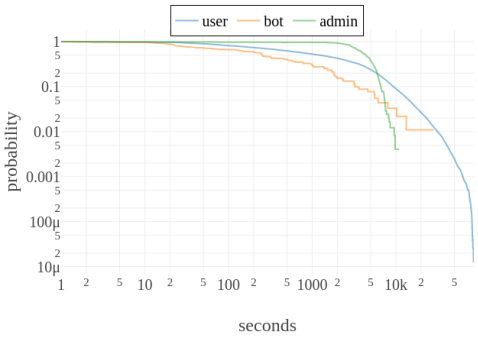
(c) br



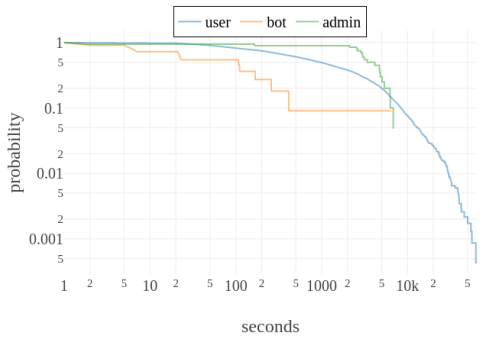
(d) ca



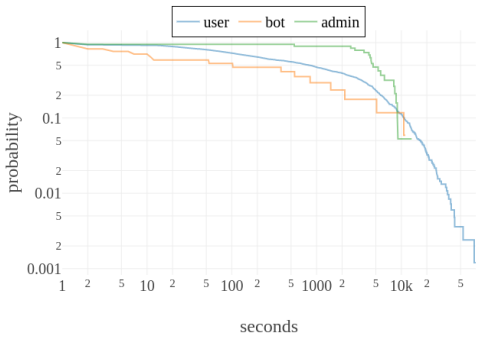
(e) cy



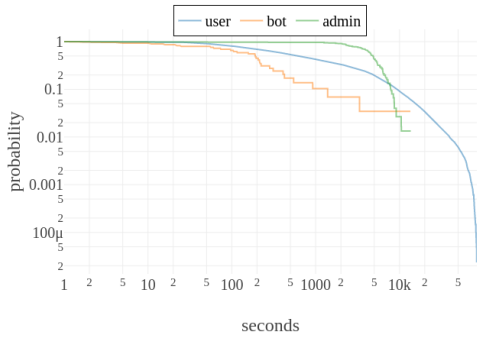
(f) de



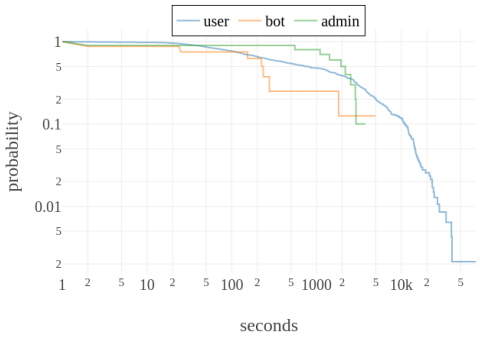
(g) el



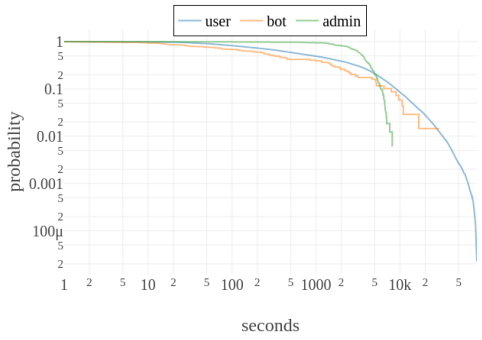
(h) eo



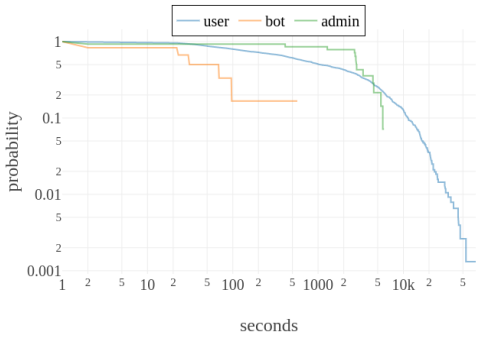
(i) es



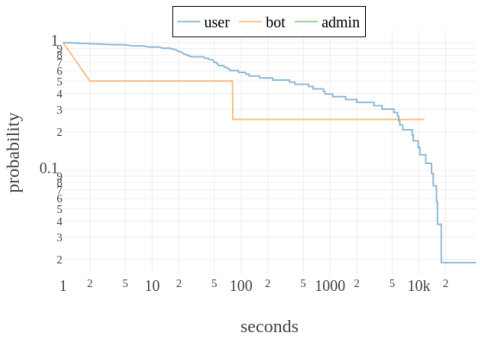
(j) eu



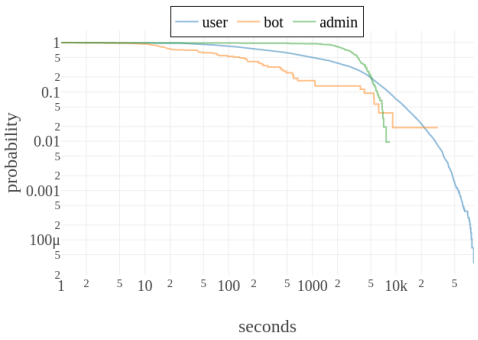
(k) fr



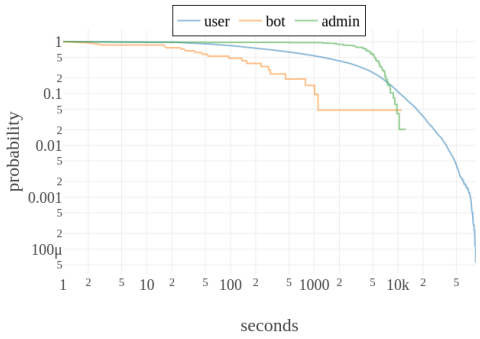
(l) gl



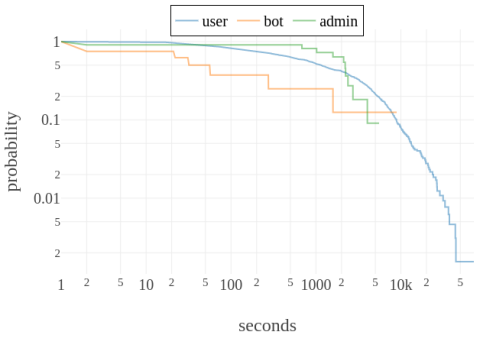
(m) ht



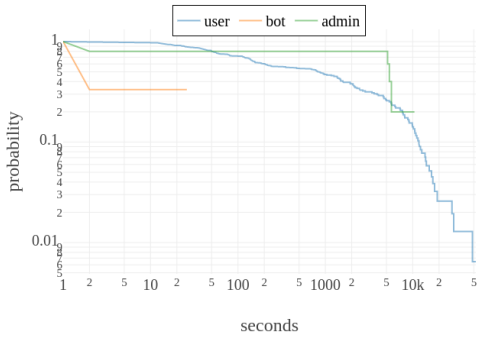
(n) it



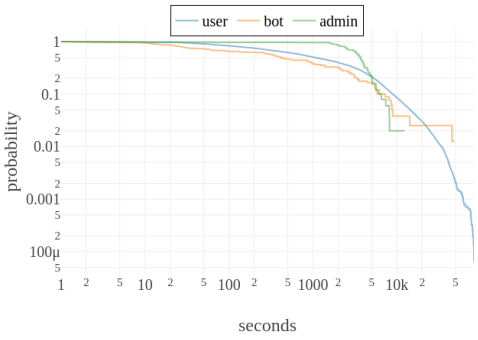
(o) ja



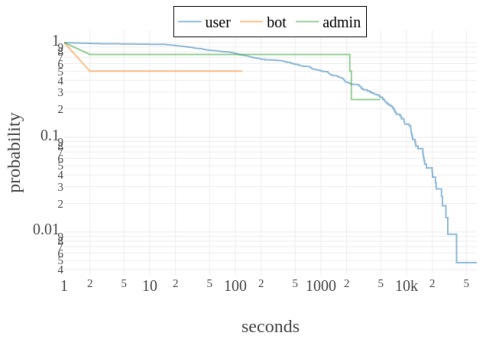
(p) lv



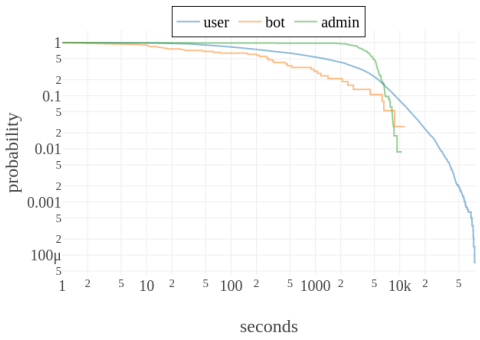
(q) nds



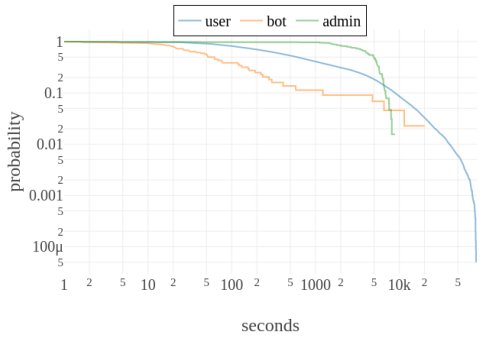
(r) nl



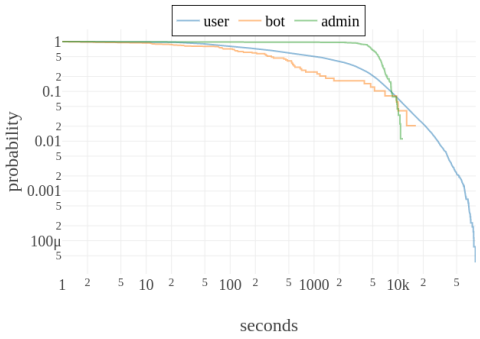
(s) oc



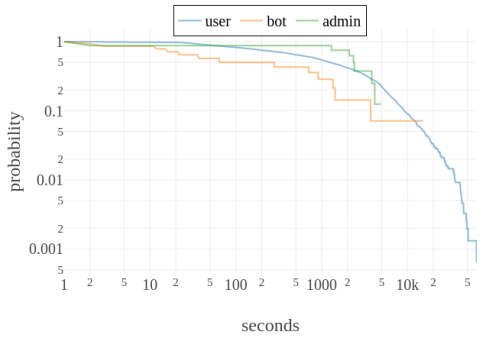
(t) pl



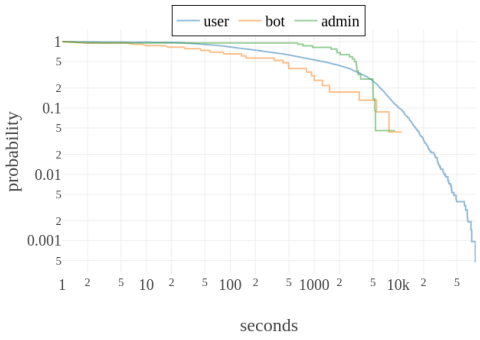
(u) pt



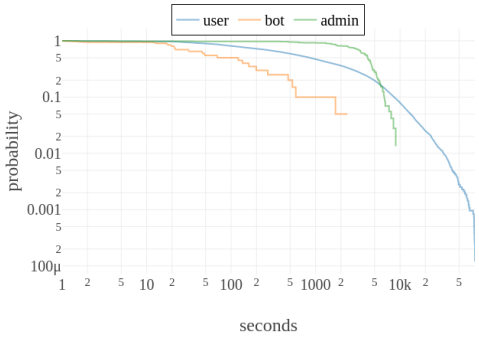
(v) ru



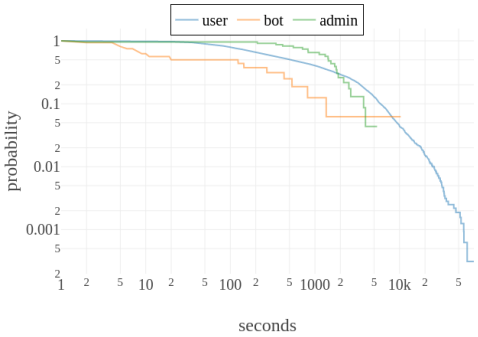
(w) sk



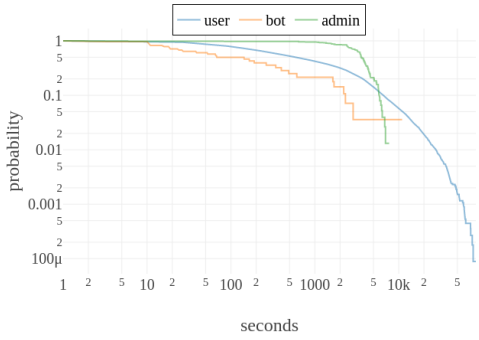
(x) sr



(y) sv



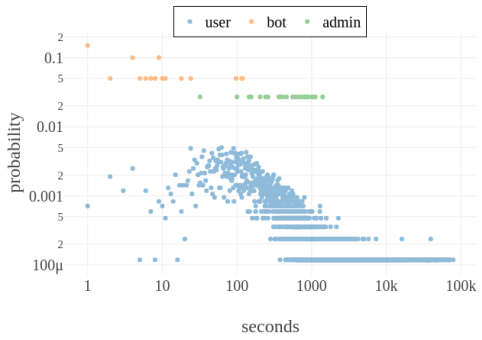
(z) vi



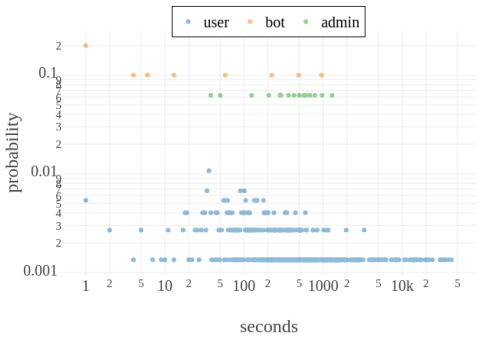
(aa) zh

Figure 37: Average user interval CCDF by role.

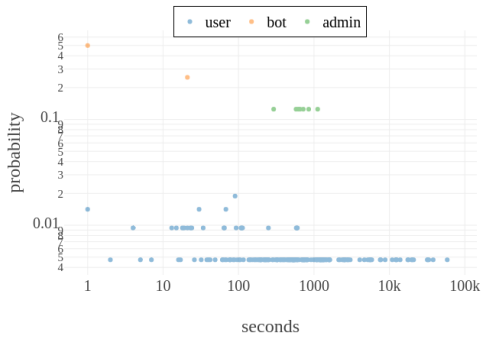
Geometric mean of user interval distribution



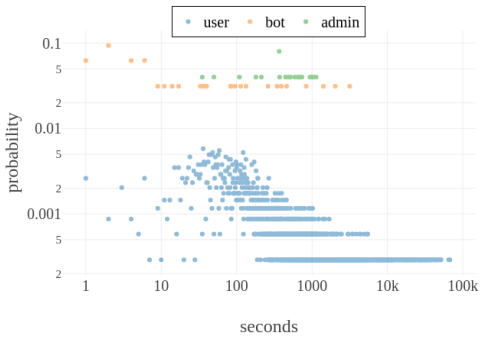
(a) ar



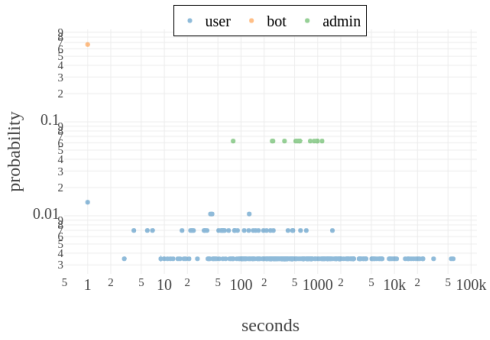
(b) bn



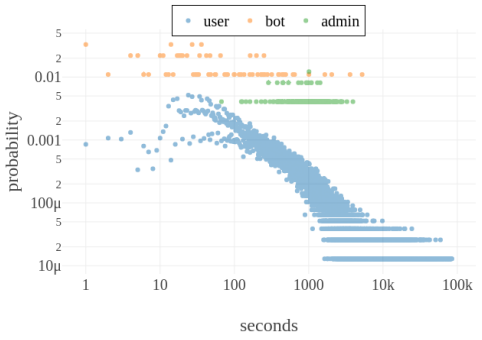
(c) br



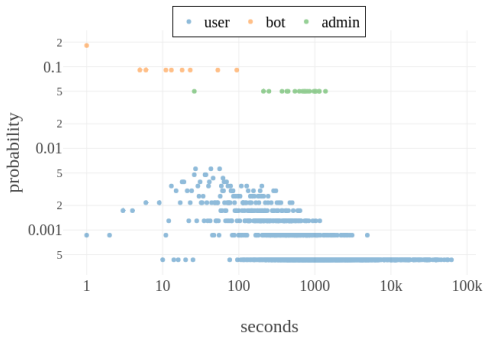
(d) ca



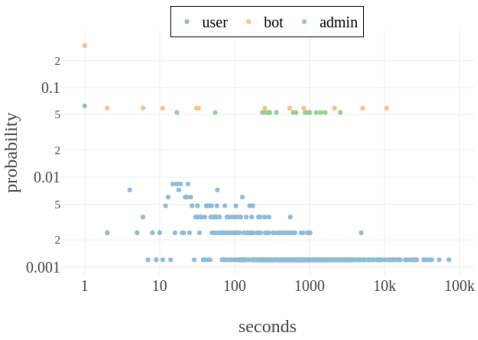
(e) cy



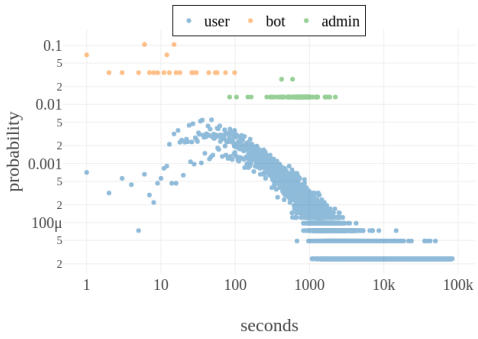
(f) de



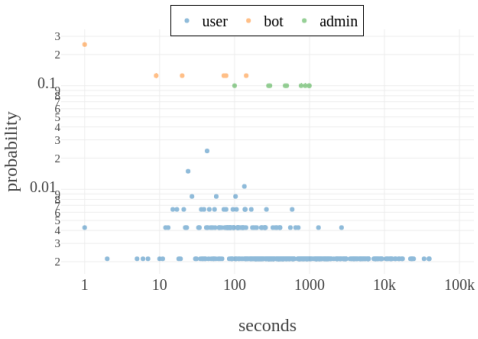
(g) el



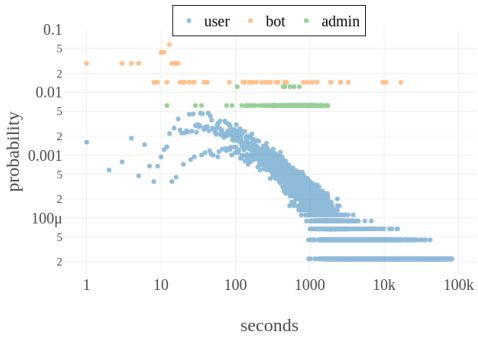
(h) eo



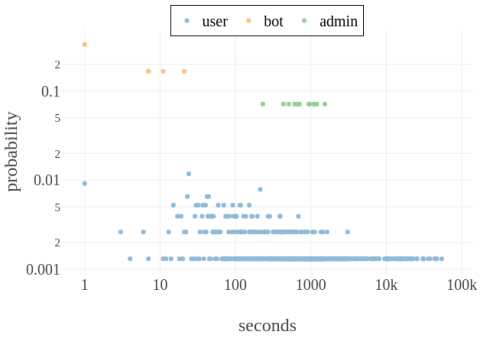
(i) es



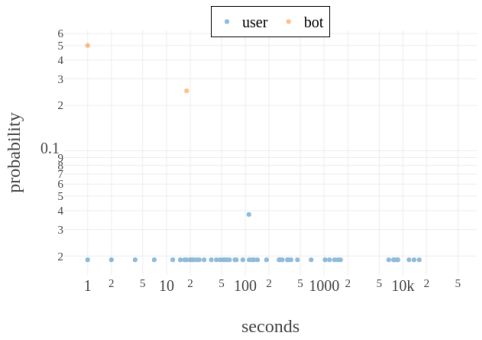
(j) eu



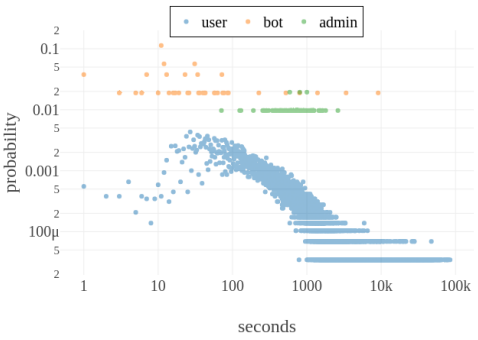
(k) fr



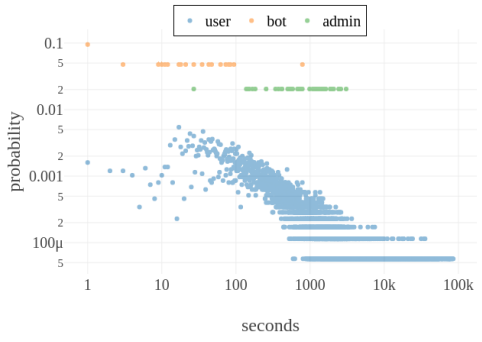
(l) gl



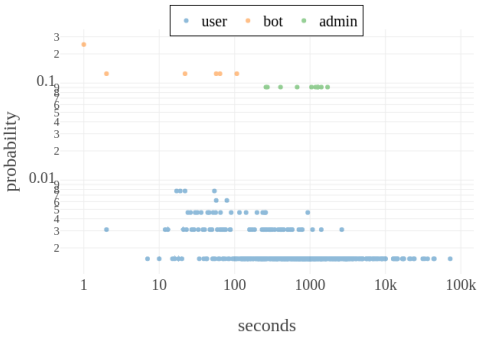
(m) ht



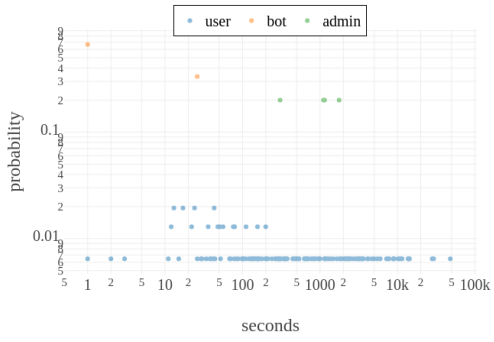
(n) it



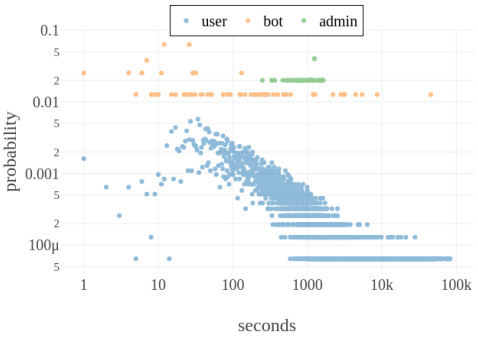
(o) ja



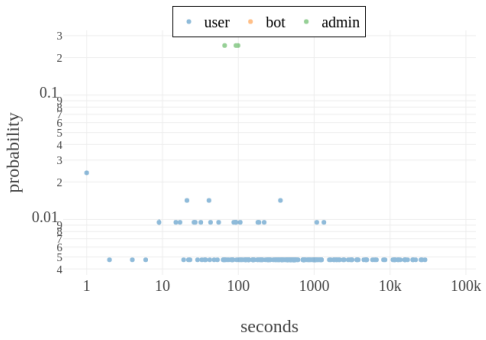
(p) lv



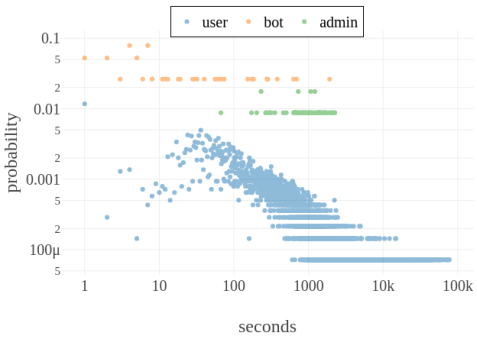
(q) nds



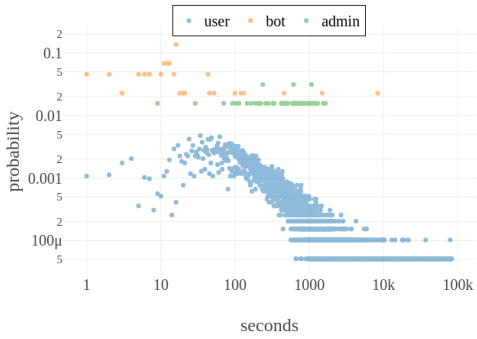
(r) nl



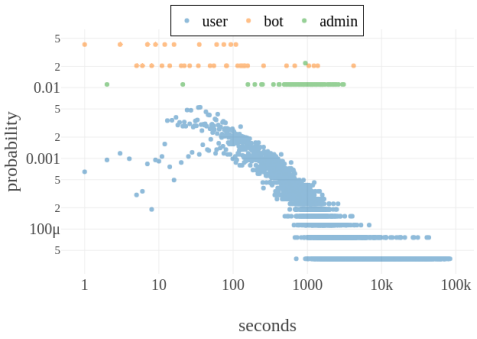
(s) oc



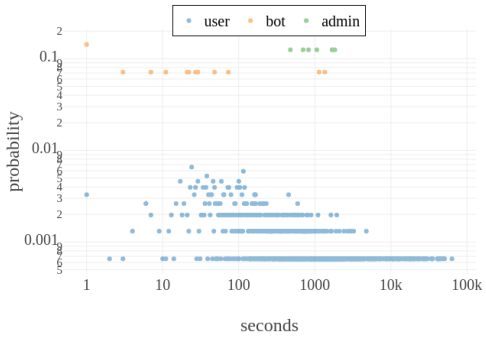
(t) pl



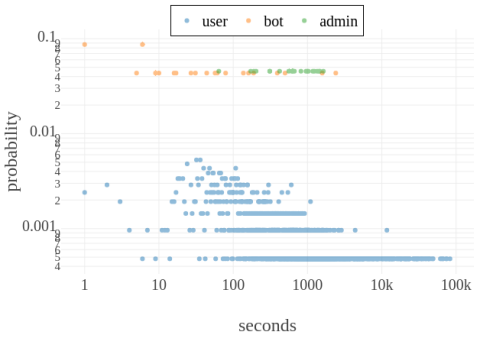
(u) pt



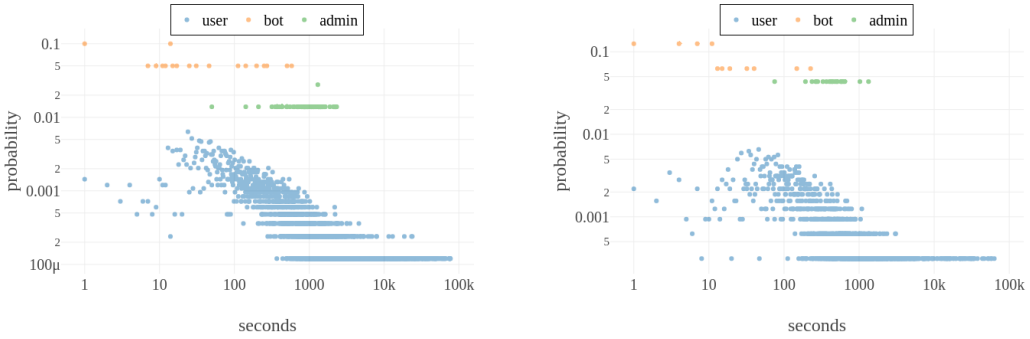
(v) ru



(w) sk



(x) sr



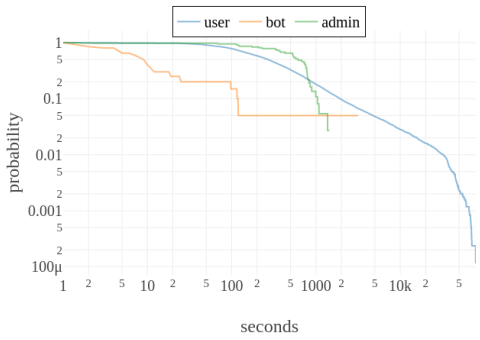
(y) sv

(z) vi

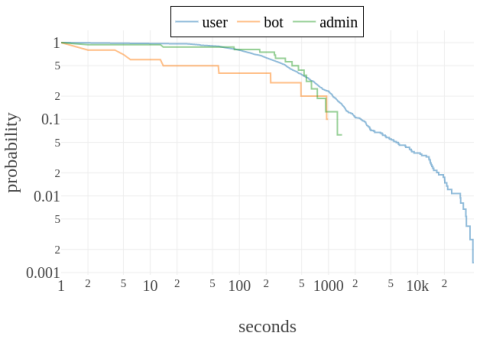
(aa) zh

Figure 38: Geometric mean of user interval distribution by role.

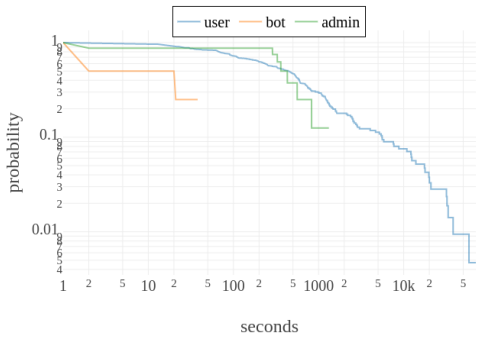
Geometric mean of user interval CCDF



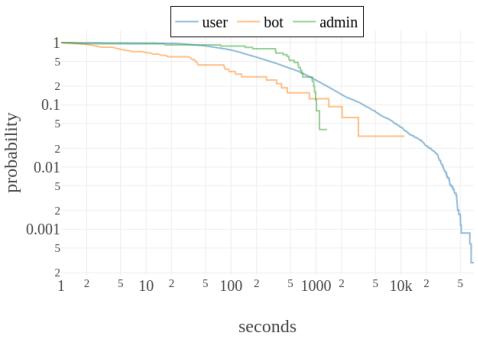
(a) ar



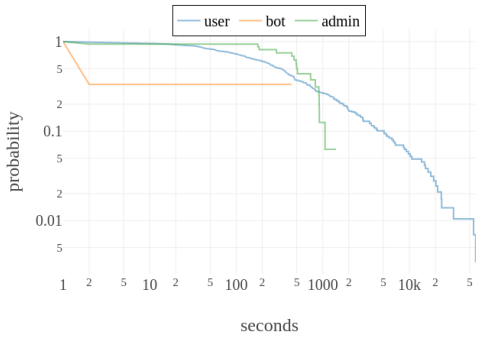
(b) bn



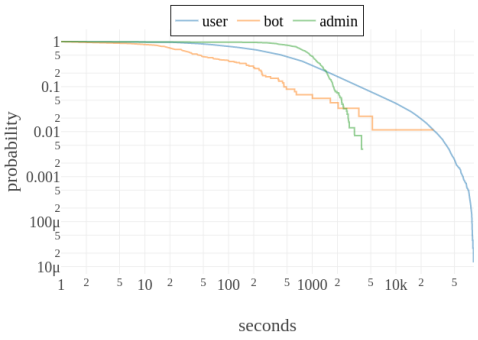
(c) br



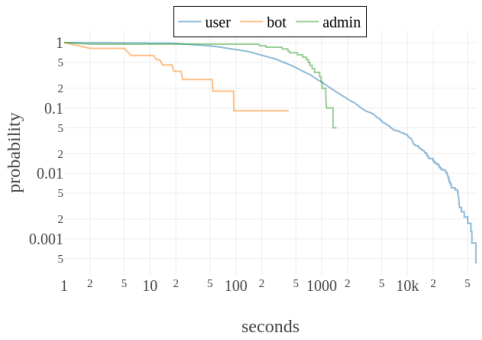
(d) ca



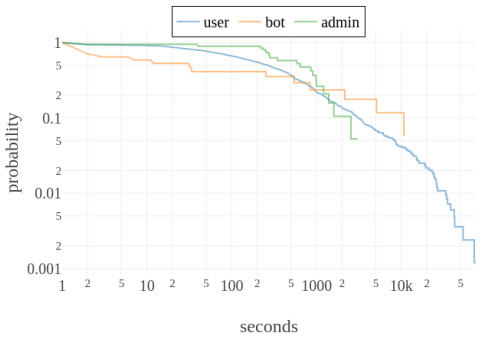
(e) cy



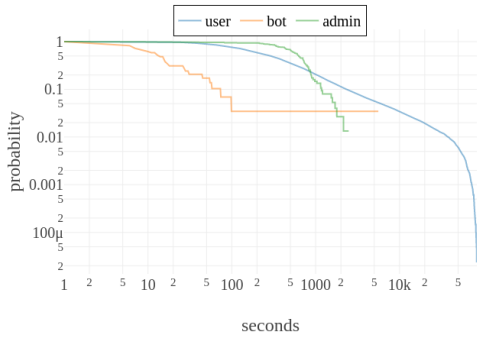
(f) de



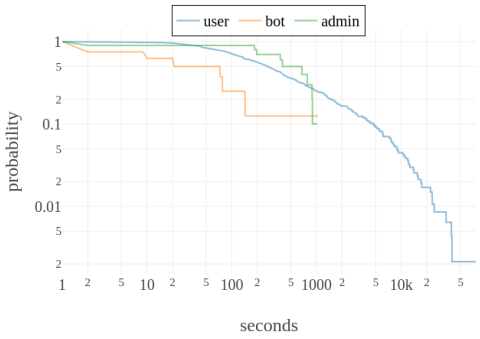
(g) el



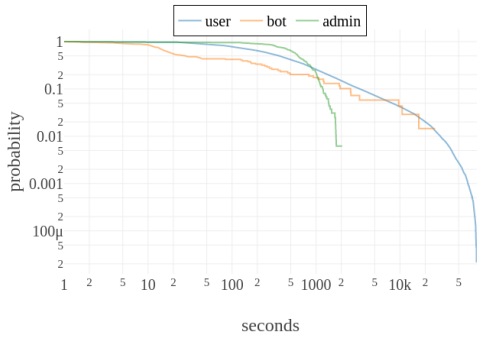
(h) eo



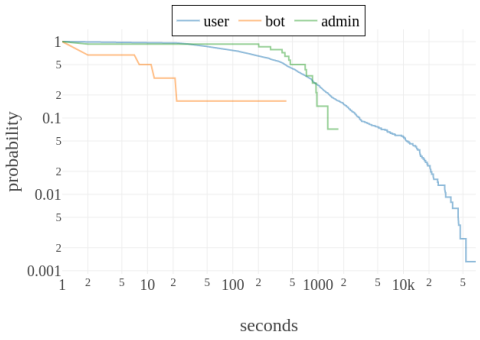
(i) es



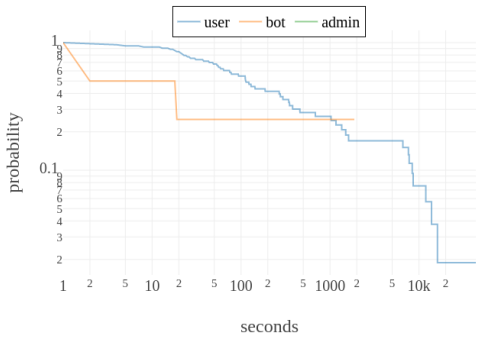
(j) eu



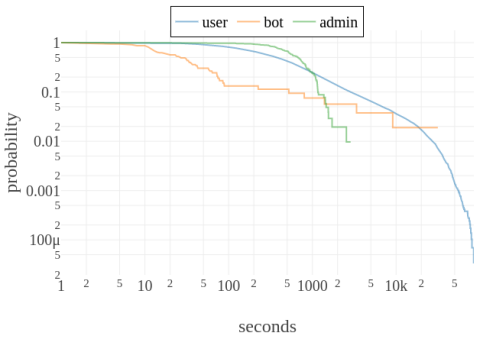
(k) fr



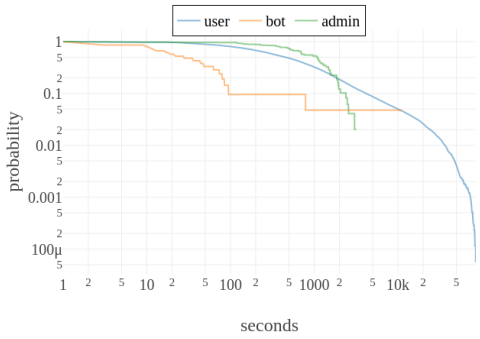
(l) gl



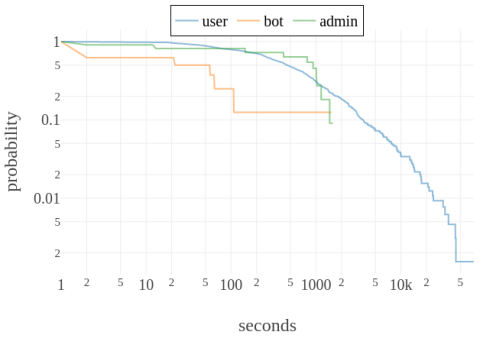
(m) ht



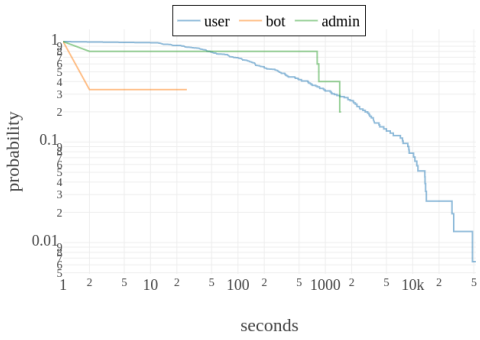
(n) it



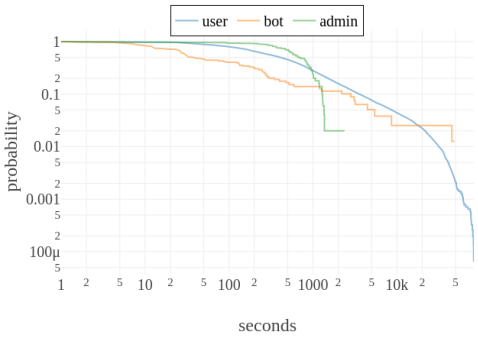
(o) ja



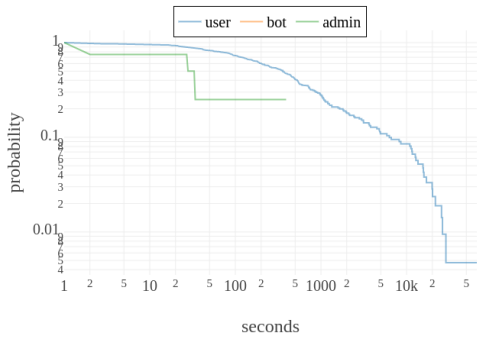
(p) lv



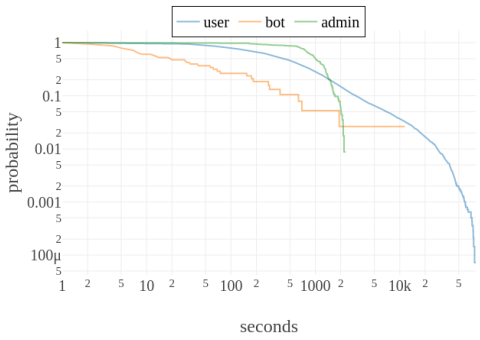
(q) nds



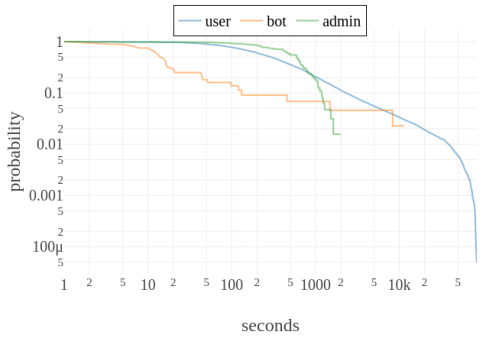
(r) nl



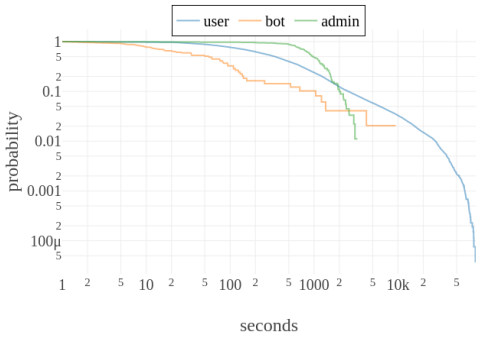
(s) oc



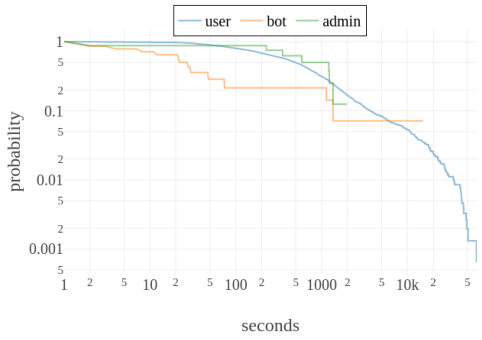
(t) pl



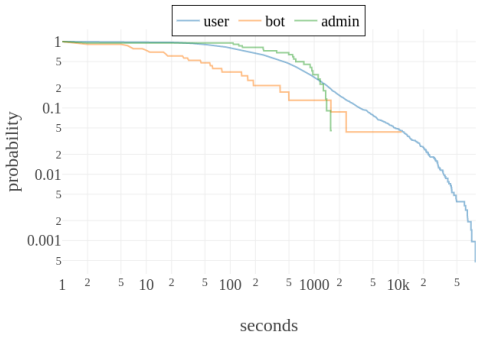
(u) pt



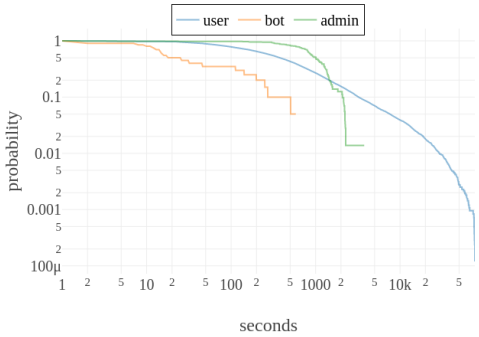
(v) ru



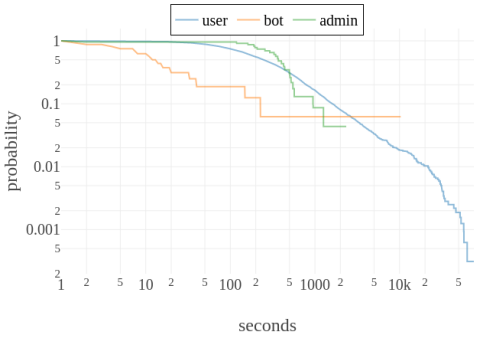
(w) sk



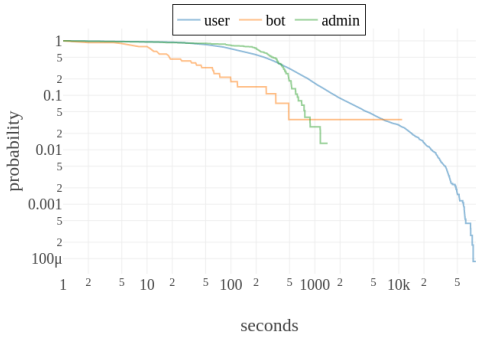
(x) sr



(y) sv



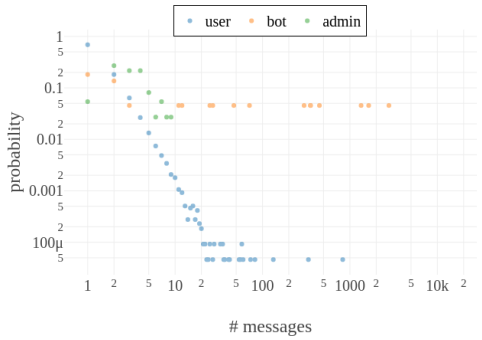
(z) vi



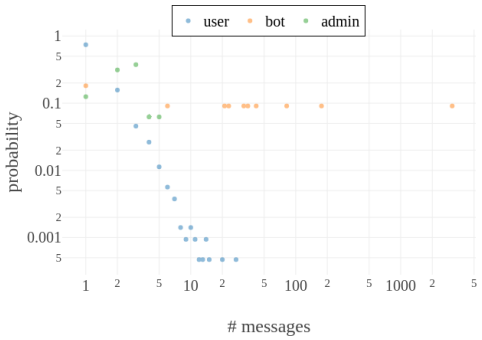
(aa) zh

Figure 39: Geometric mean of user interval CCDF by role.

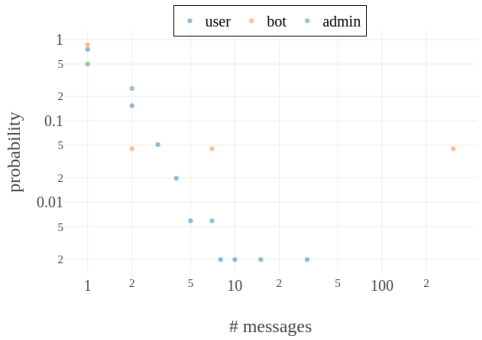
Arithmetic mean of messages per day



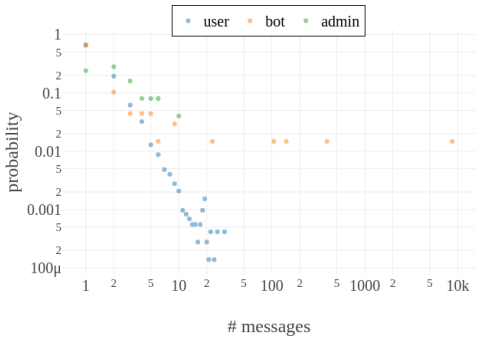
(a) ar



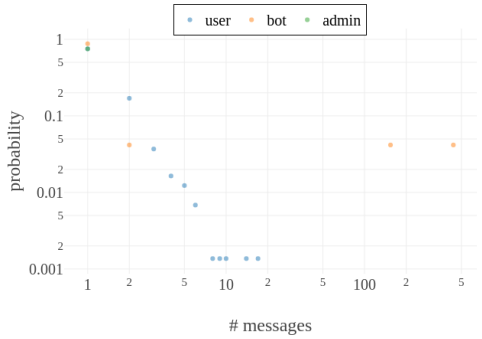
(b) bn



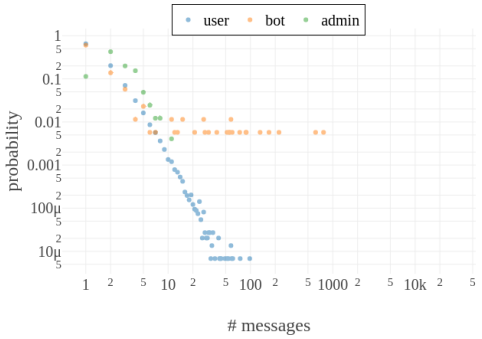
(c) br



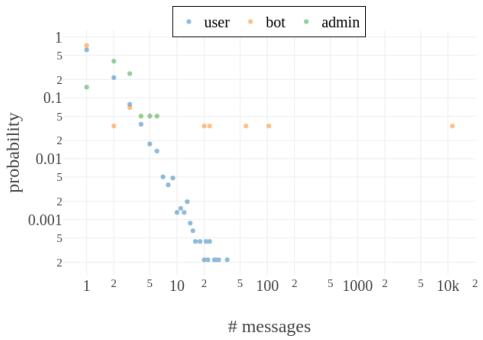
(d) ca



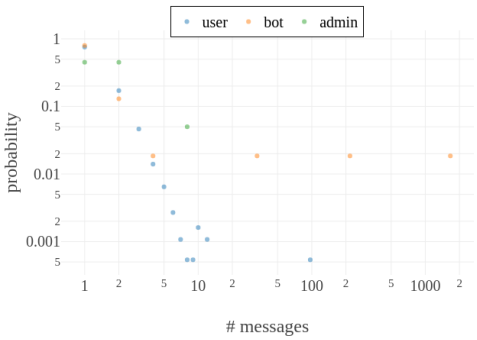
(e) cy



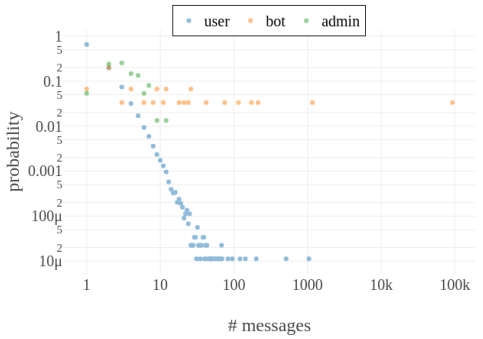
(f) de



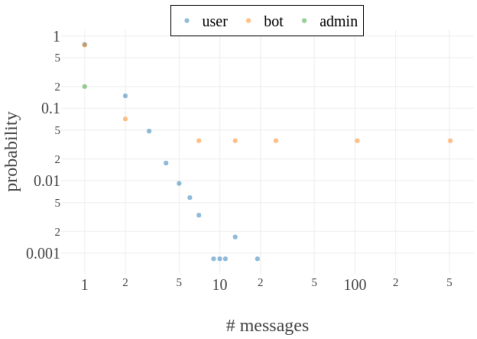
(g) el



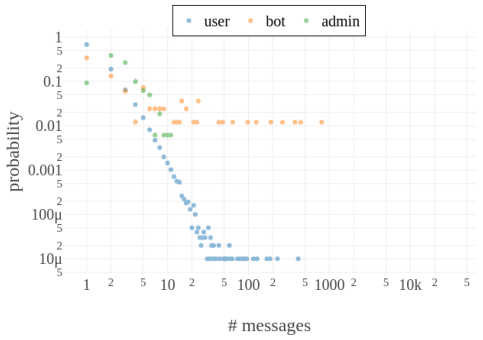
(h) eo



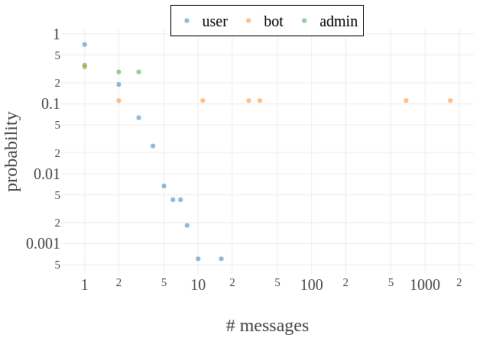
(i) es



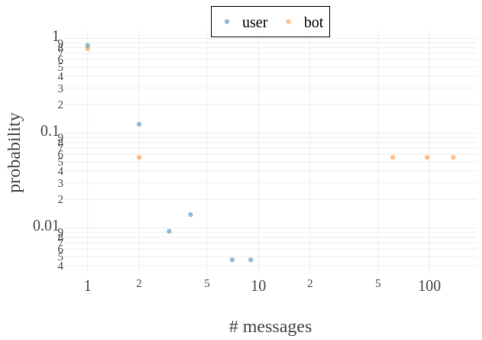
(j) eu



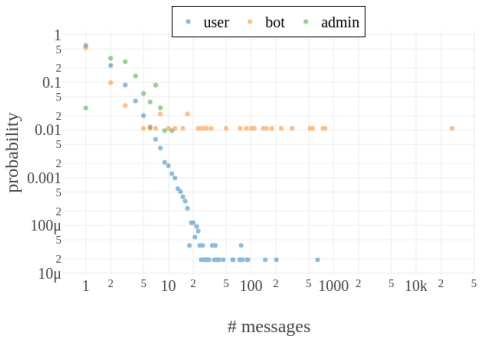
(k) fr



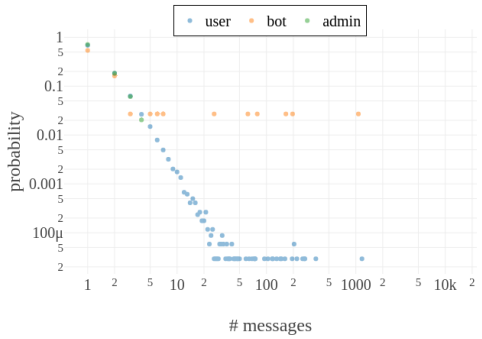
(l) gl



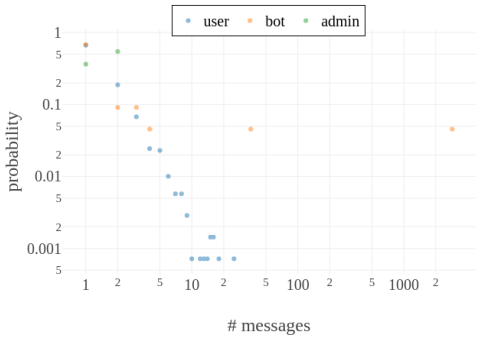
(m) ht



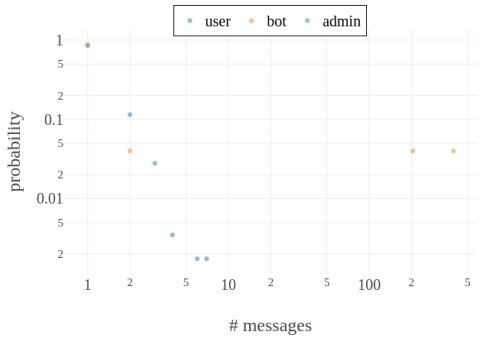
(n) it



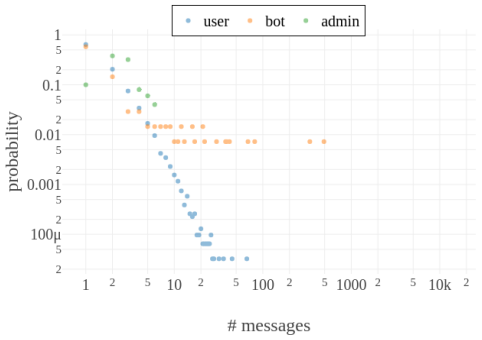
(o) ja



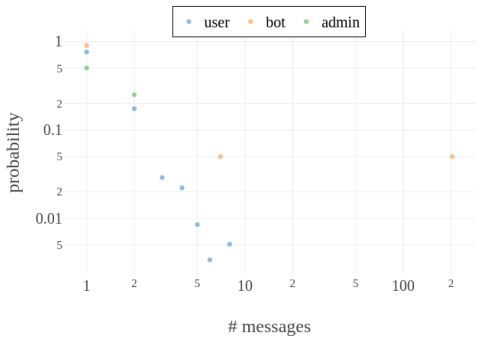
(p) lv



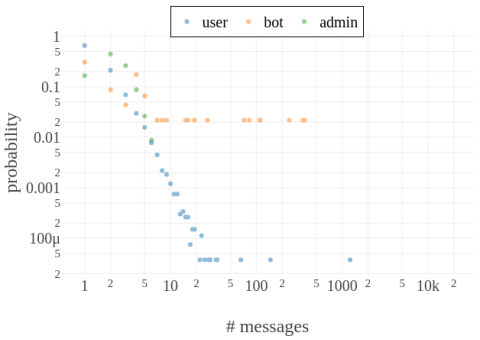
(q) nds



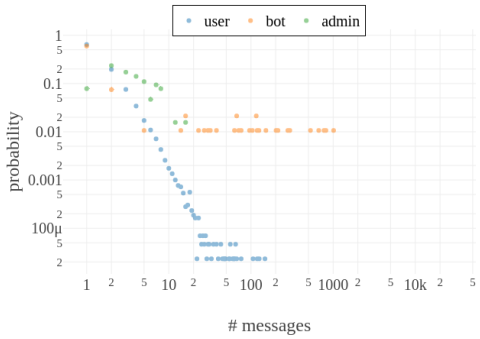
(r) nl



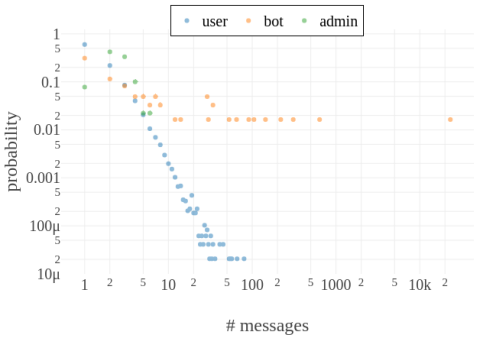
(s) oc



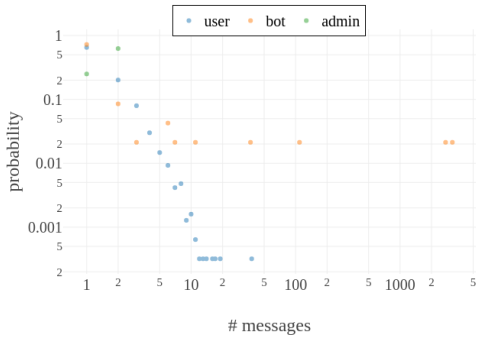
(t) pl



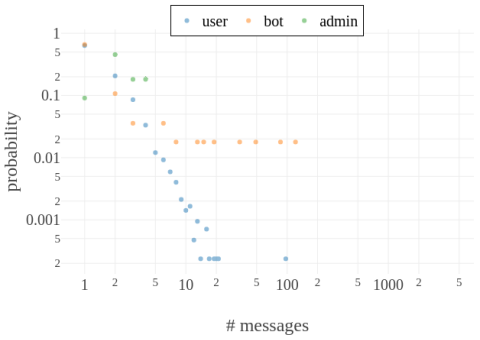
(u) pt



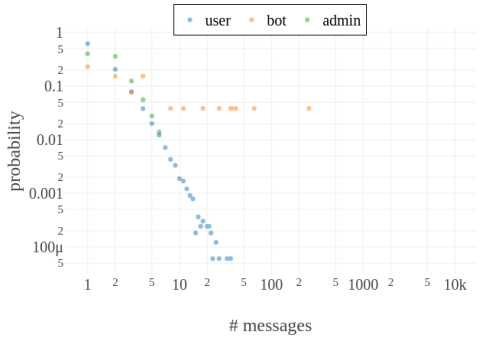
(v) ru



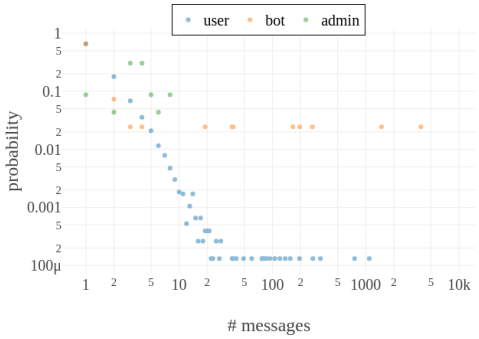
(w) sk



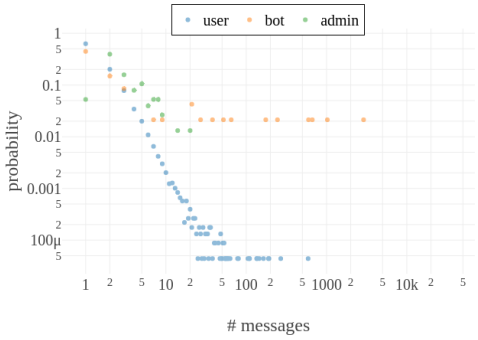
(x) sr



(y) sv



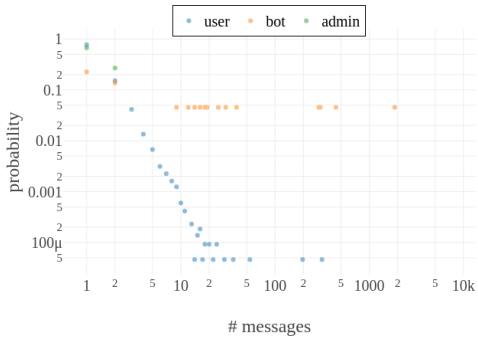
(z) vi



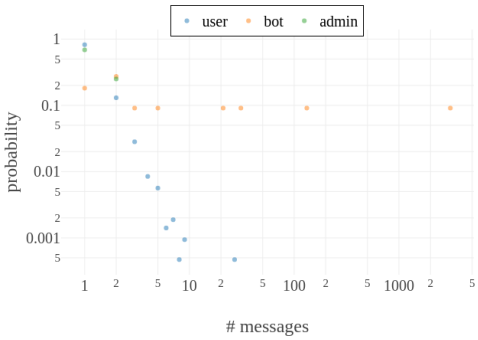
(aa) zh

Figure 40: Average message rate per day.

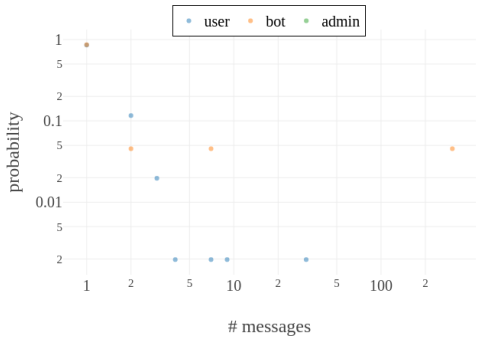
Arithmetic mean of messages per hour



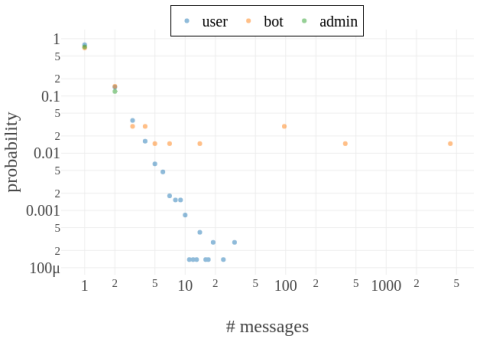
(a) ar



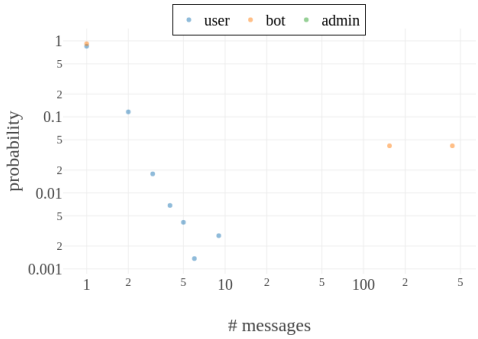
(b) bn



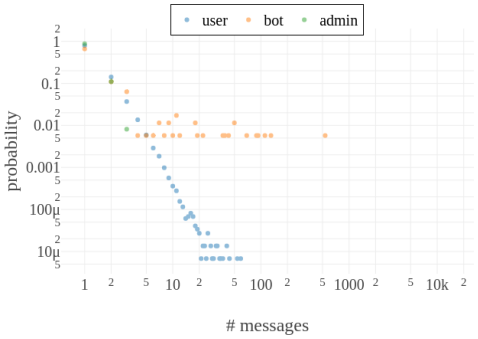
(c) br



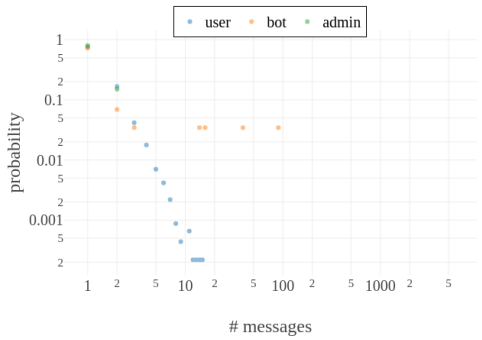
(d) ca



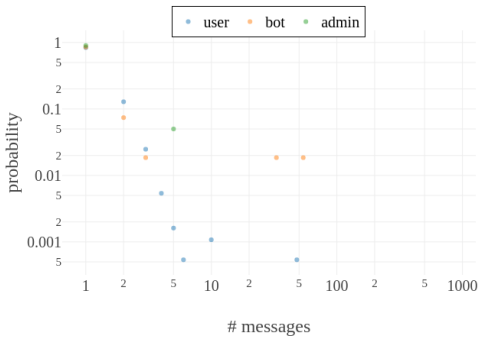
(e) cy



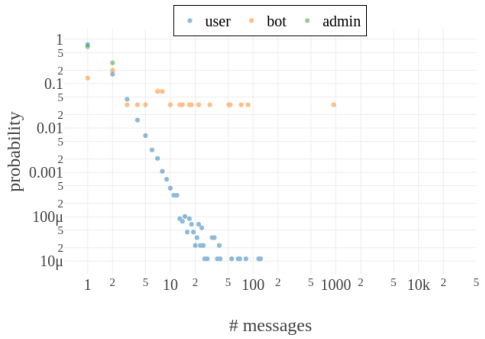
(f) de



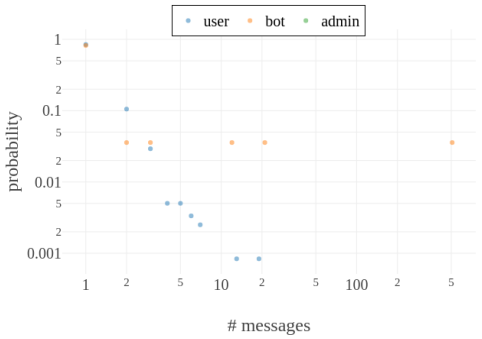
(g) el



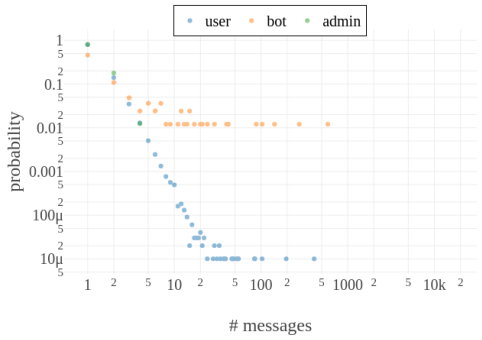
(h) eo



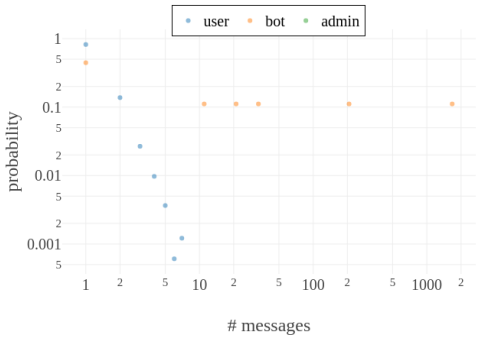
(i) es



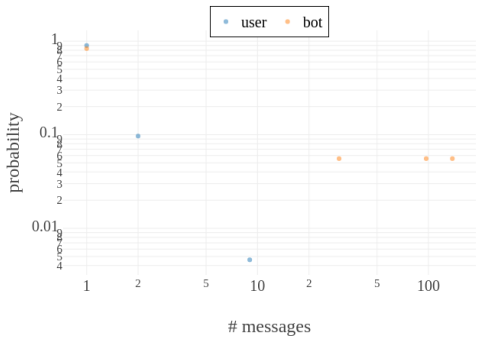
(j) eu



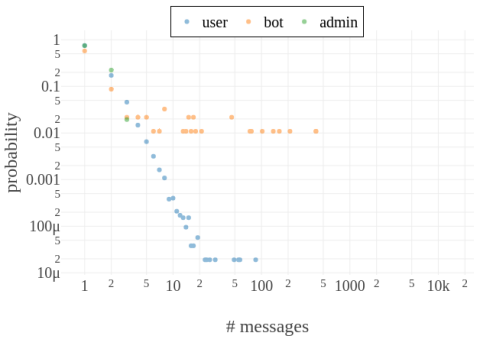
(k) fr



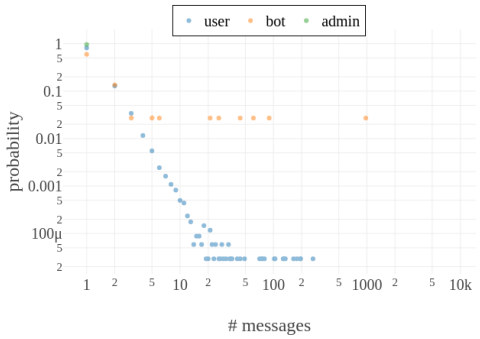
(l) gl



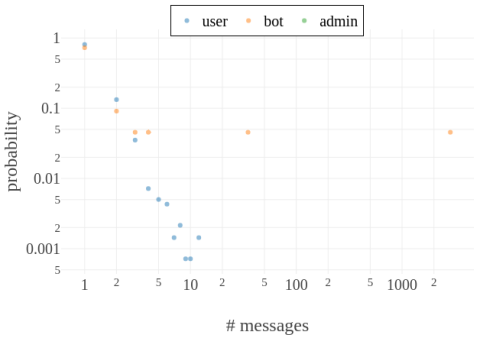
(m) ht



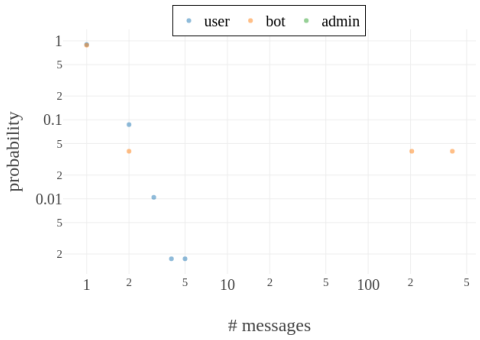
(n) it



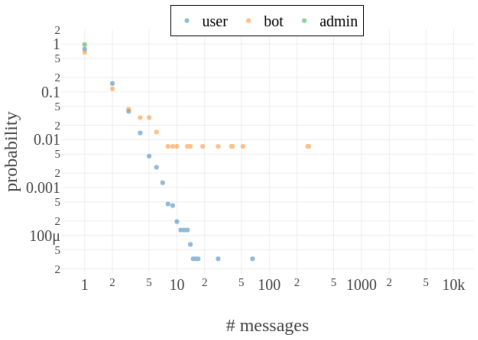
(o) ja



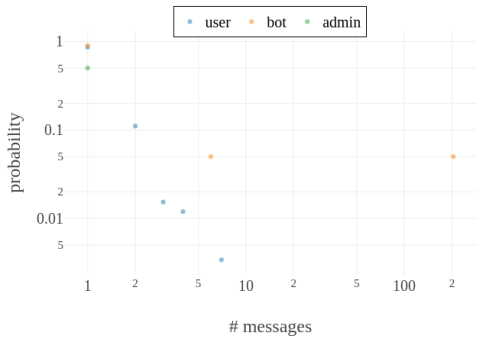
(p) lv



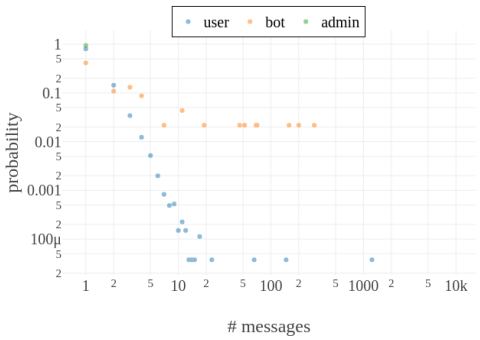
(q) nds



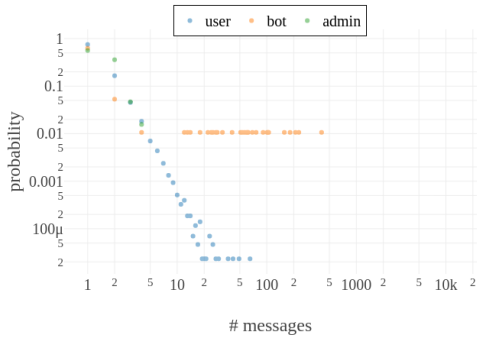
(r) nl



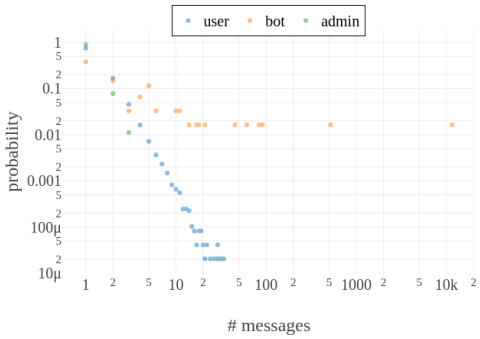
(s) oc



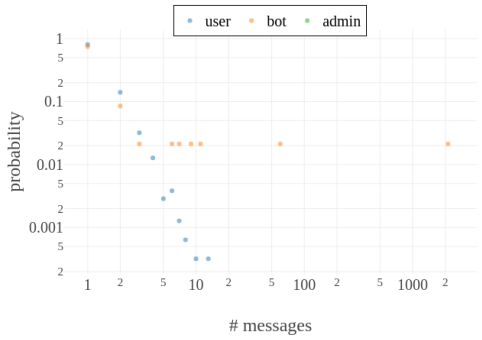
(t) pl



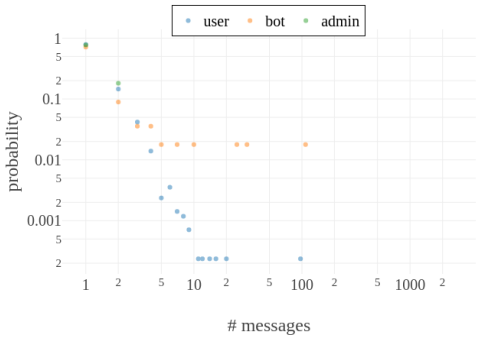
(u) pt



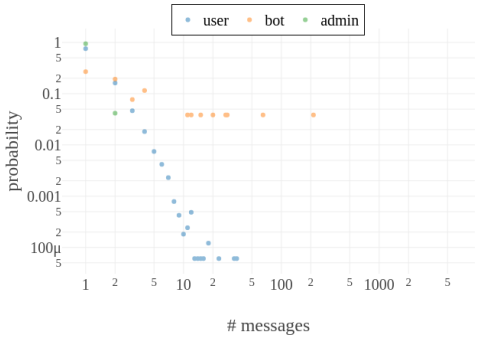
(v) ru



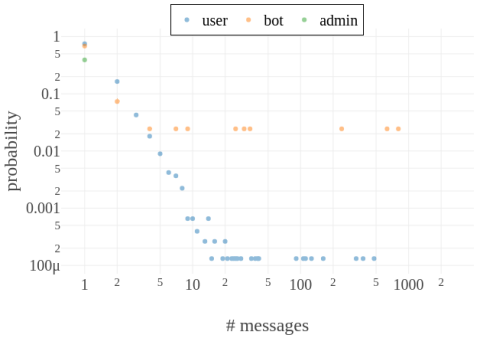
(w) sk



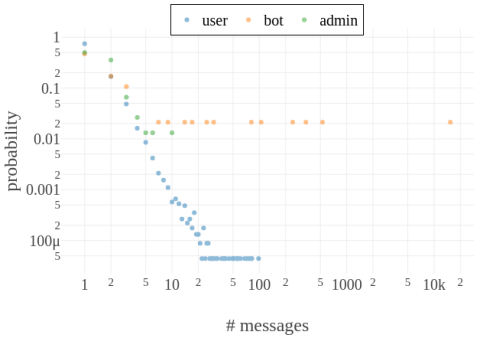
(x) sr



(y) sv



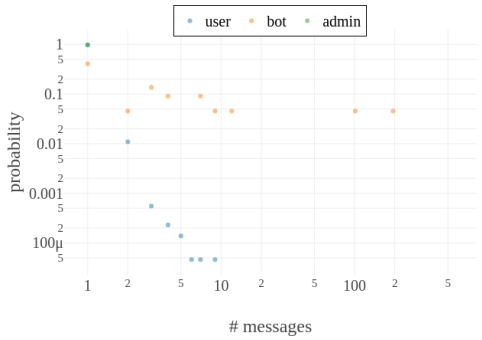
(z) vi



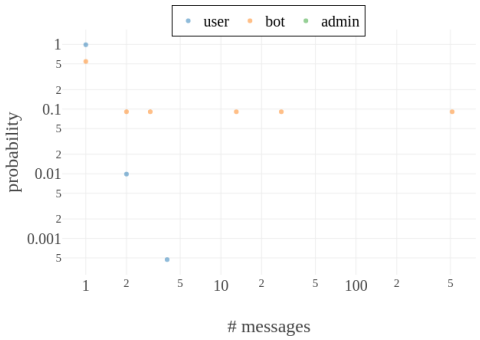
(aa) zh

Figure 41: Average message rate per hour.

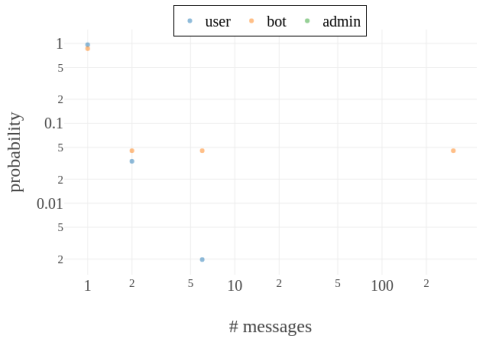
Arithmetic mean of messages per minute



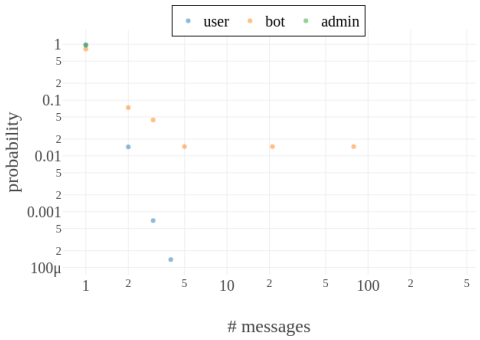
(a) ar



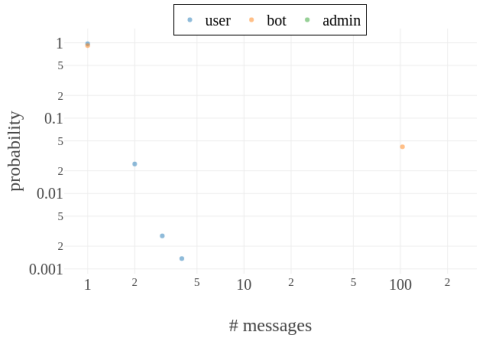
(b) bn



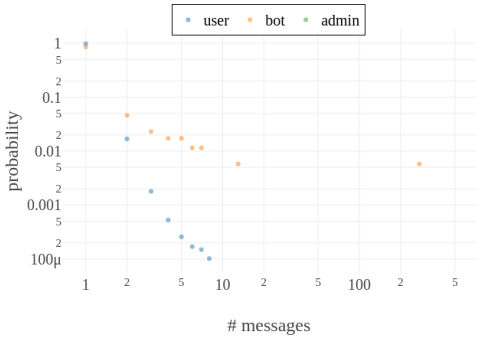
(c) br



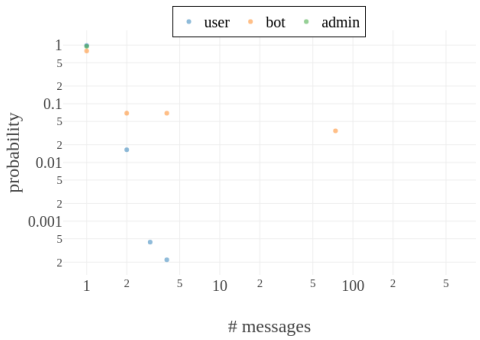
(d) ca



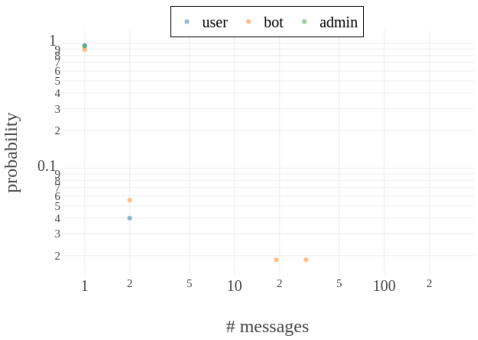
(e) cy



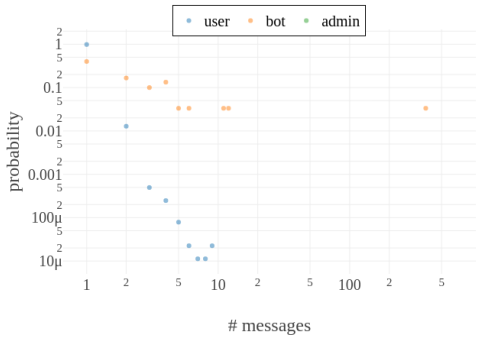
(f) de



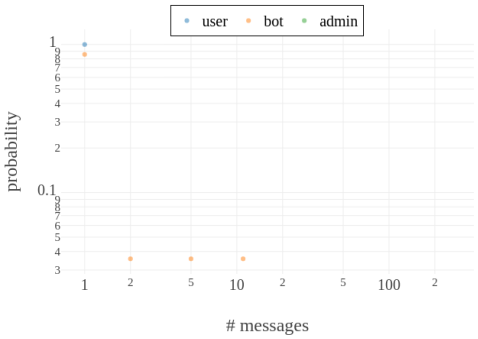
(g) el



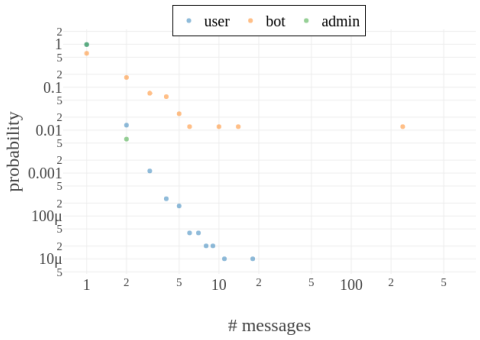
(h) eo



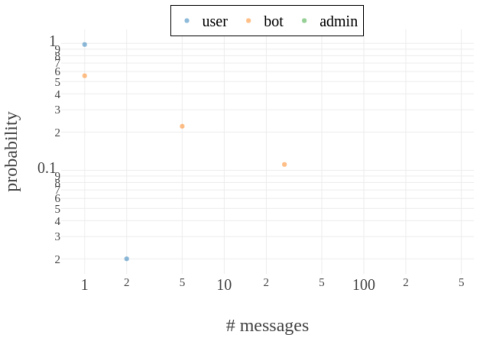
(i) es



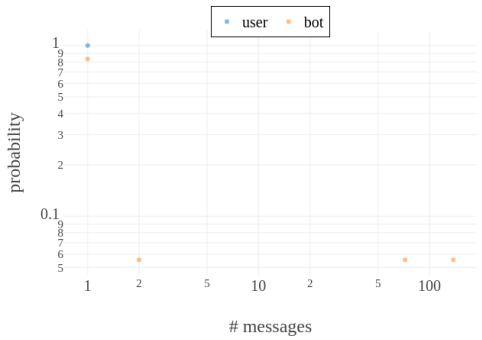
(j) eu



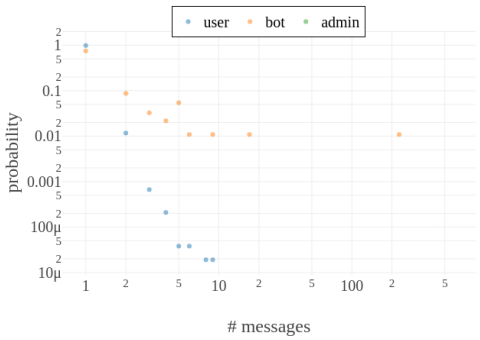
(k) fr



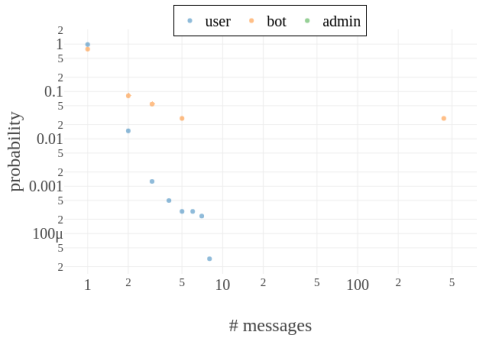
(l) gl



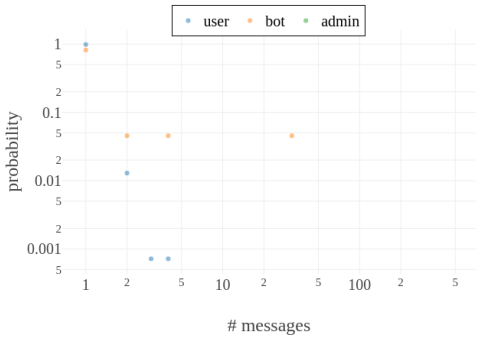
(m) ht



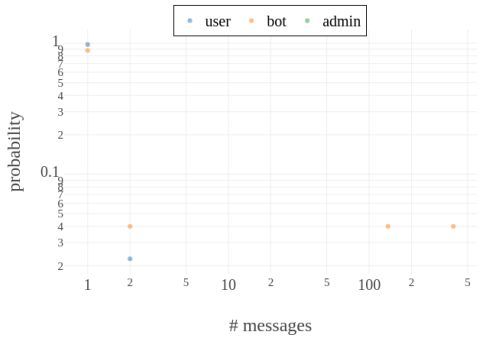
(n) it



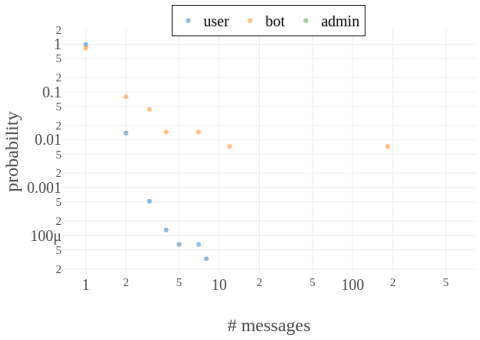
(o) ja



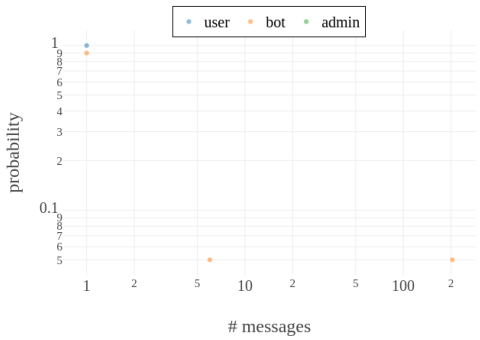
(p) lv



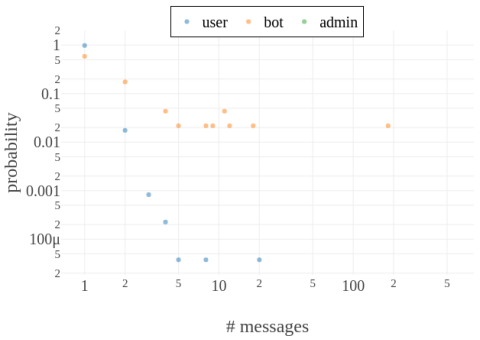
(q) nds



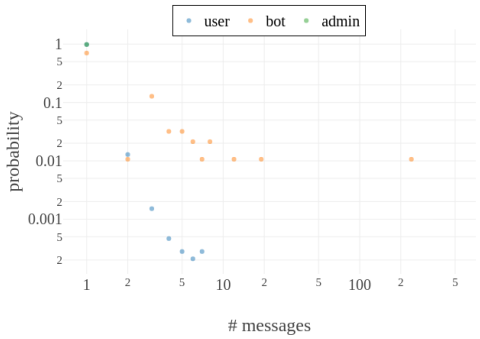
(r) nl



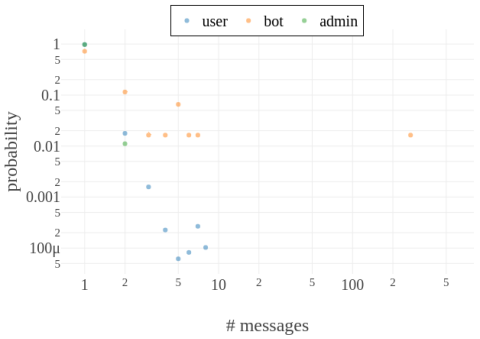
(s) oc



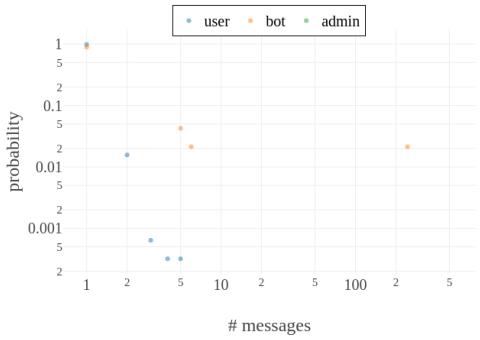
(t) pl



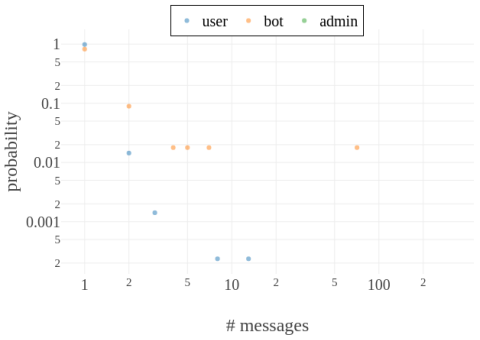
(u) pt



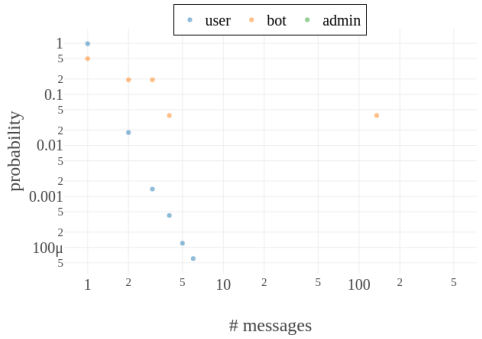
(v) ru



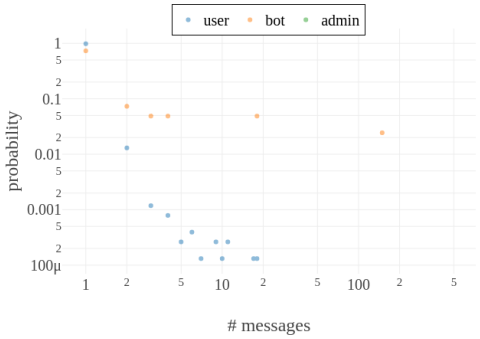
(w) sk



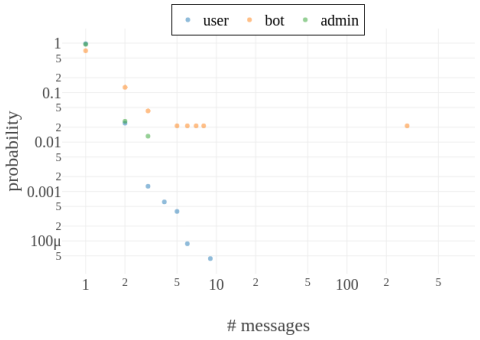
(x) sr



(y) sv



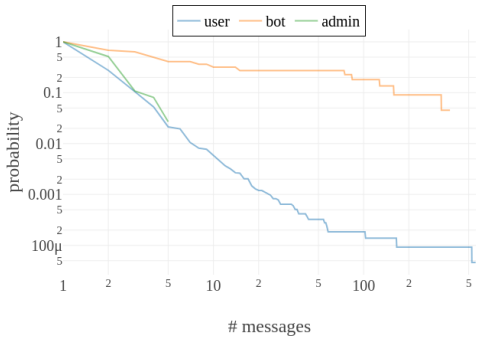
(z) vi



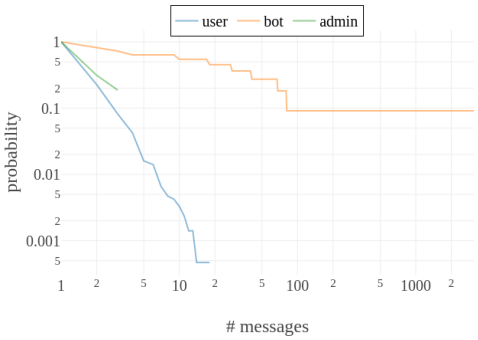
(aa) zh

Figure 42: Average message rate per minute.

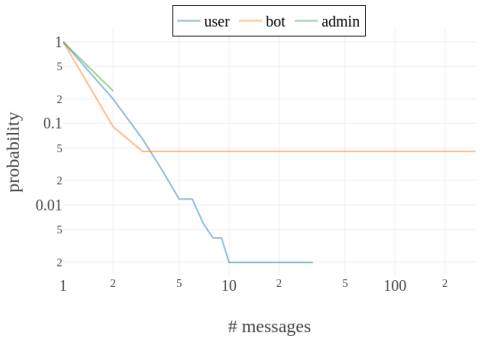
Geometric mean of messages per day



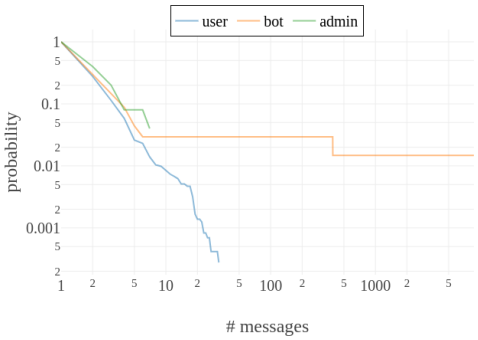
(a) ar



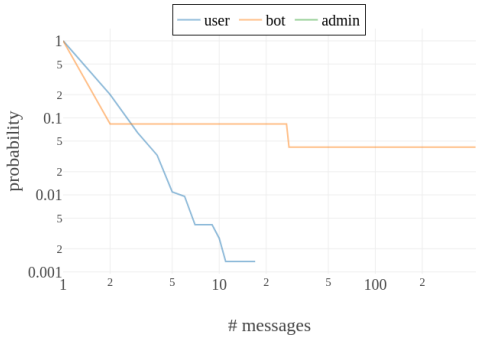
(b) bn



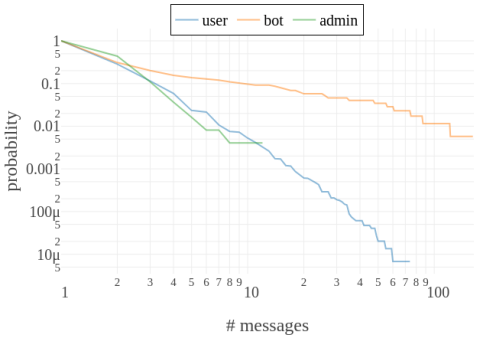
(c) br



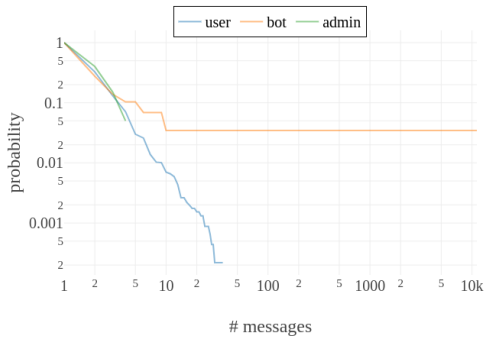
(d) ca



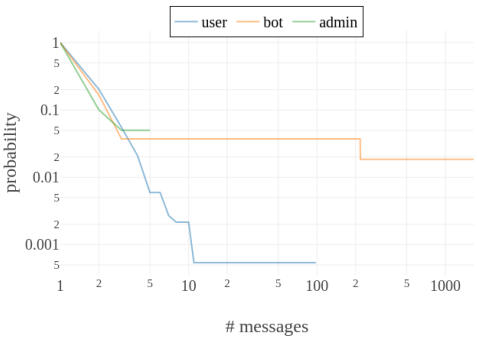
(e) cy



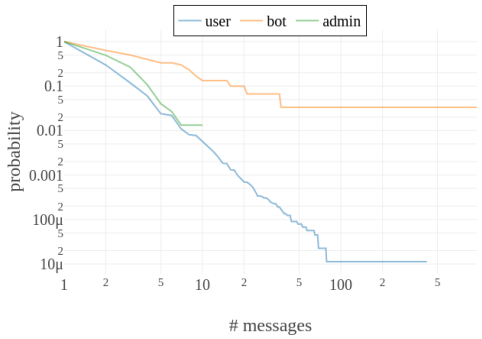
(f) de



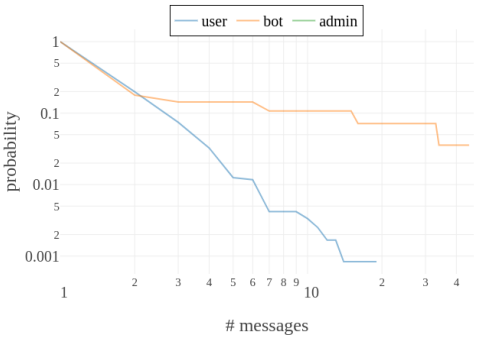
(g) el



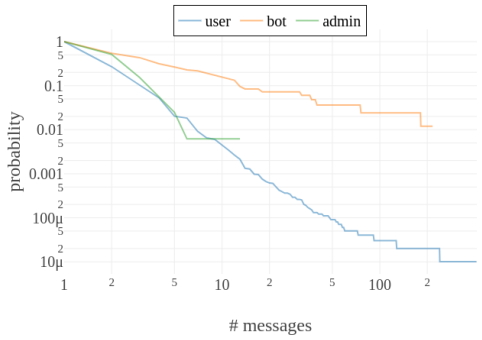
(h) eo



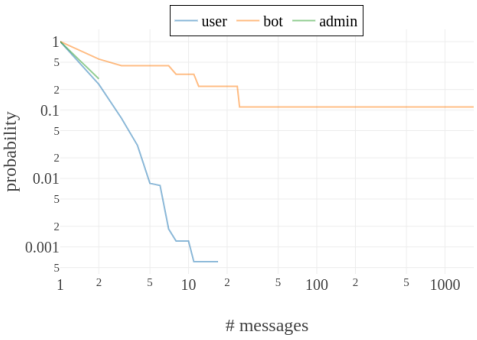
(i) es



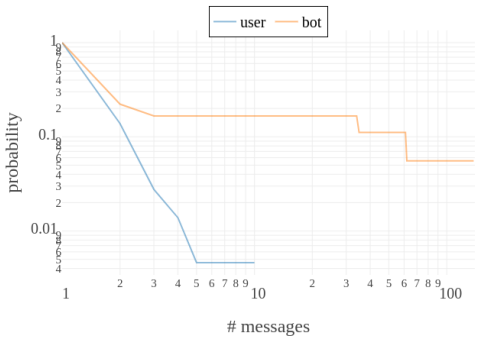
(j) eu



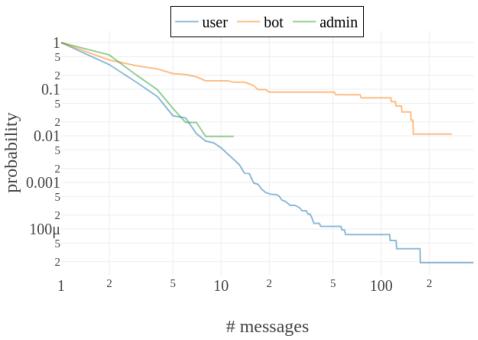
(k) fr



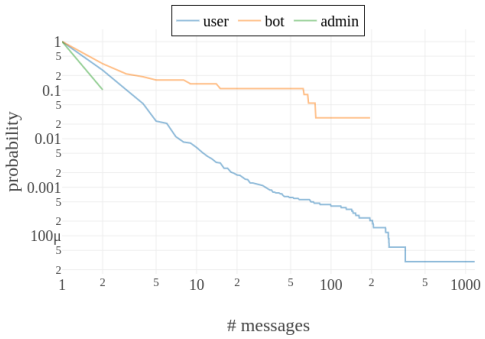
(l) gl



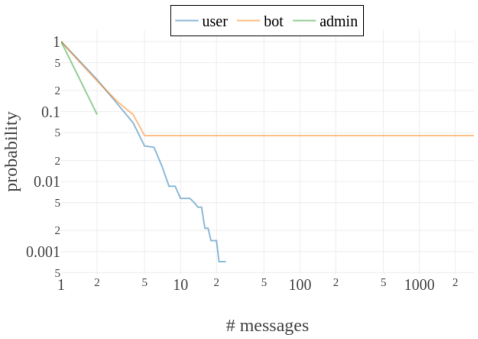
(m) ht



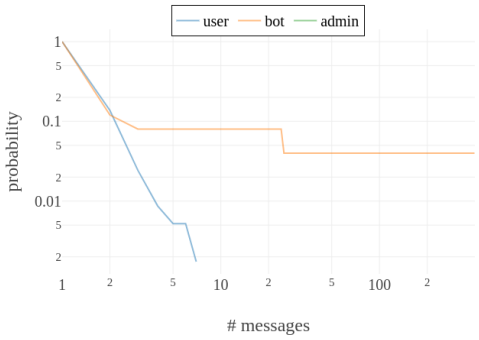
(n) it



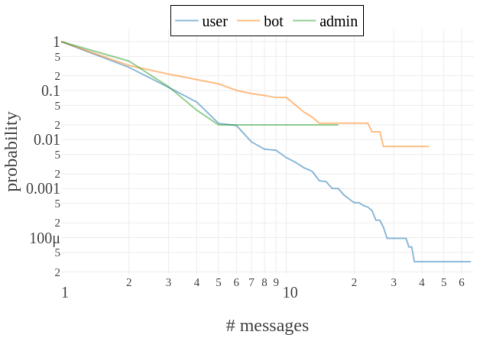
(o) ja



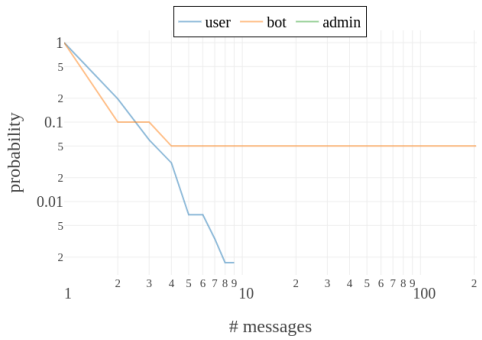
(p) lv



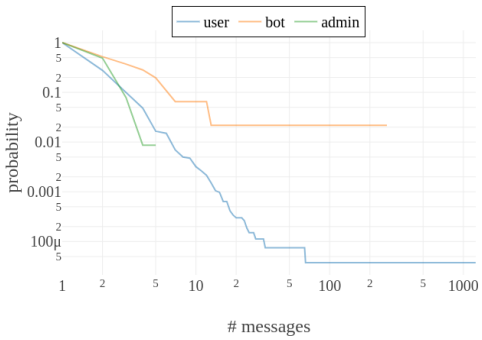
(q) nds



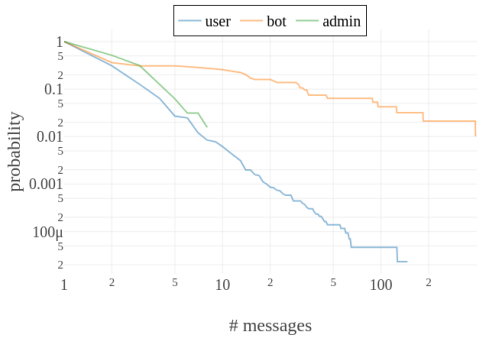
(r) nl



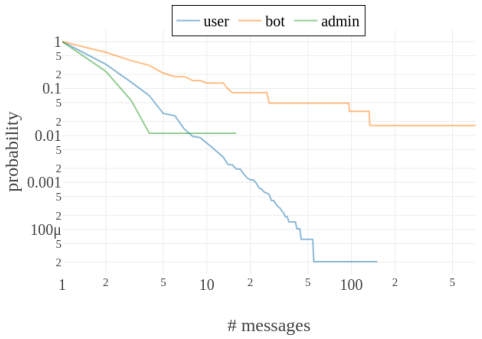
(s) oc



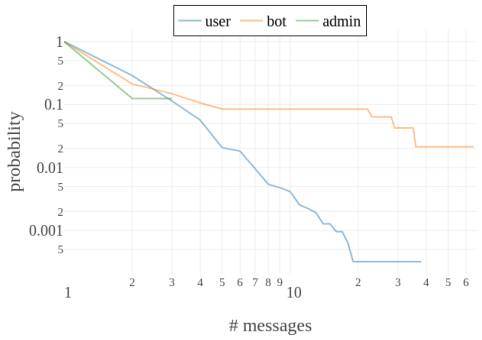
(t) pl



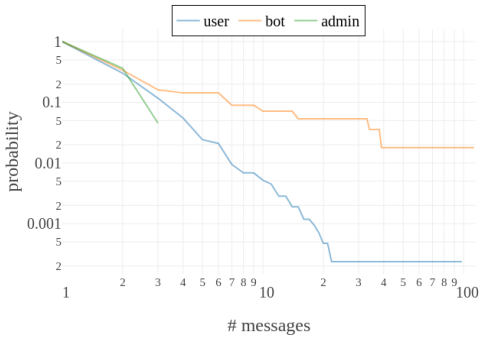
(u) pt



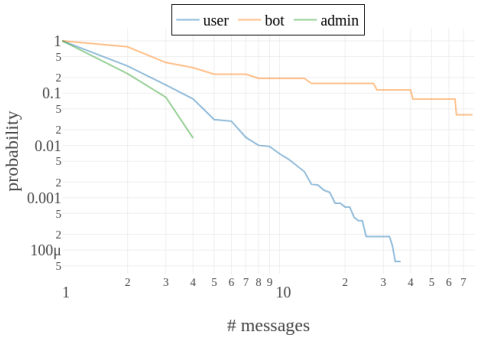
(v) ru



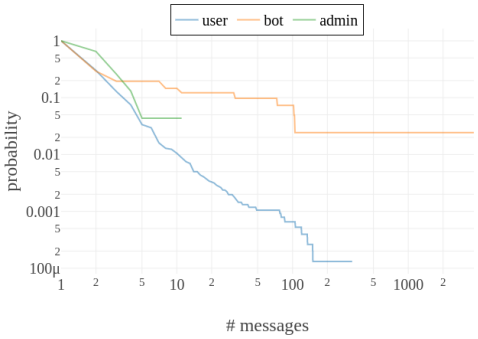
(w) sk



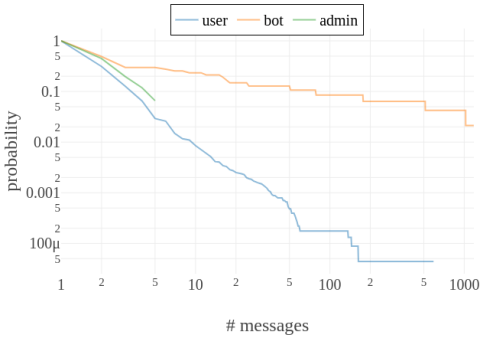
(x) sr



(y) sv



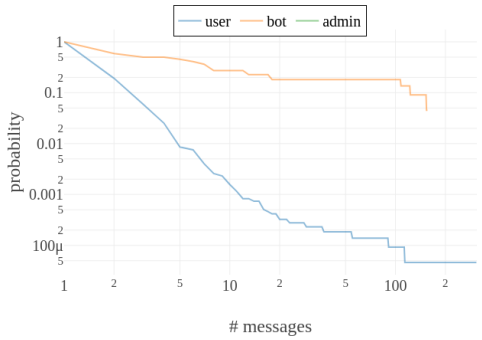
(z) vi



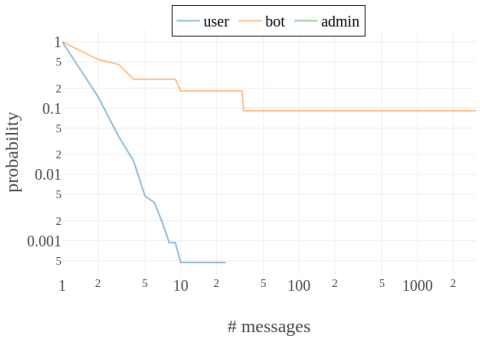
(aa) zh

Figure 43: Geometric mean of messages per day.

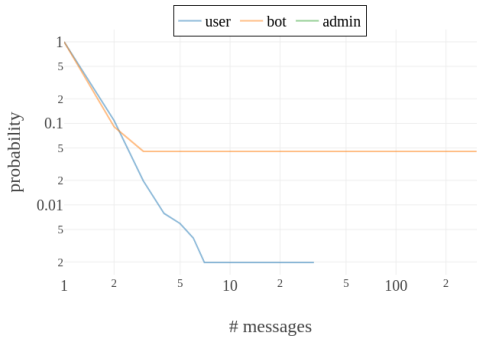
Geometric mean of messages per hour



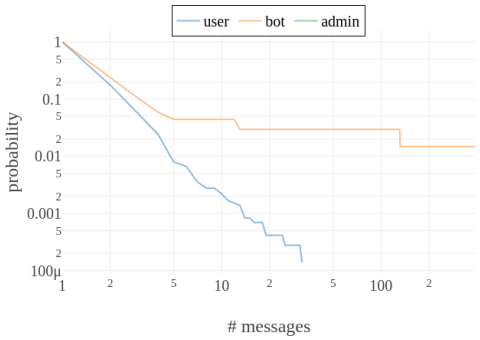
(a) ar



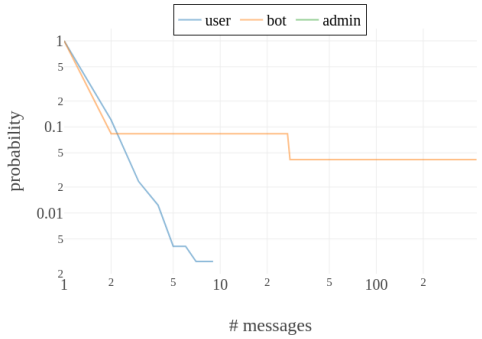
(b) bn



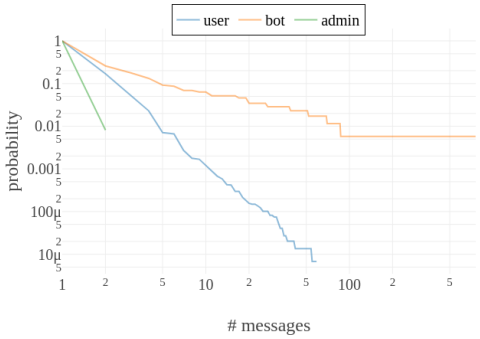
(c) br



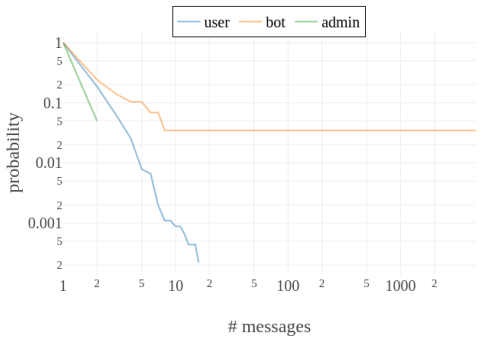
(d) ca



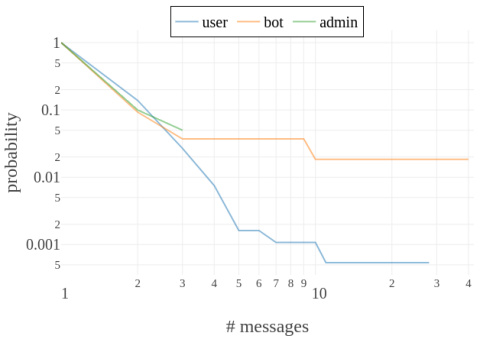
(e) cy



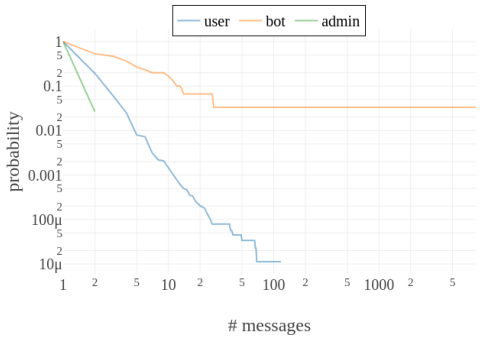
(f) de



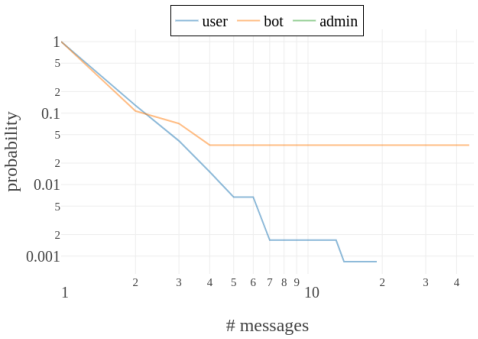
(g) el



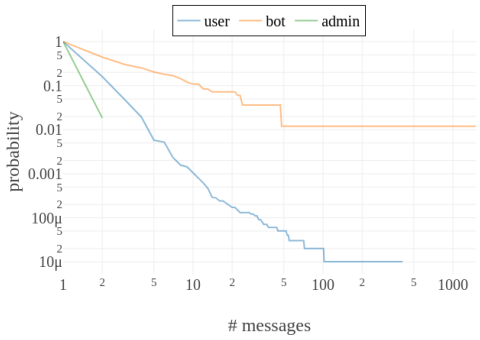
(h) eo



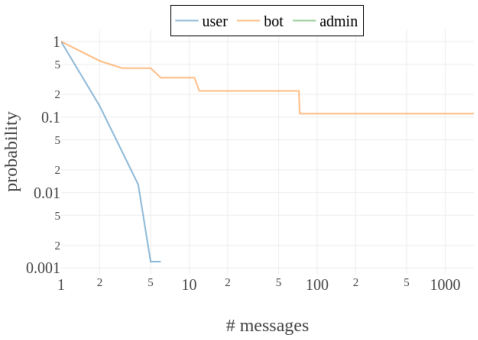
(i) es



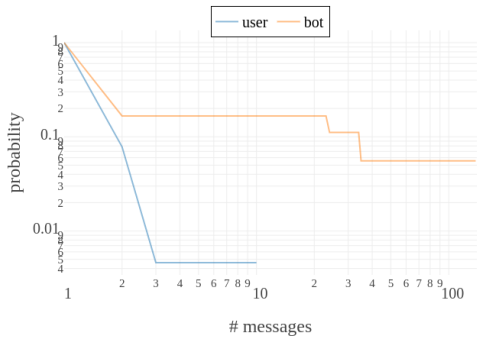
(j) eu



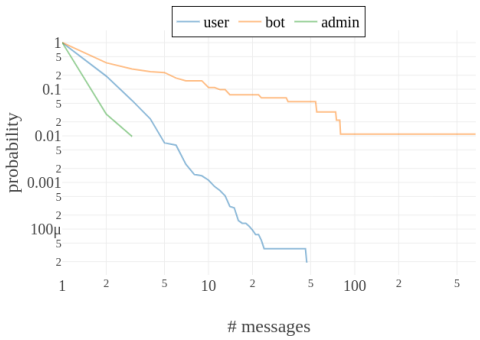
(k) fr



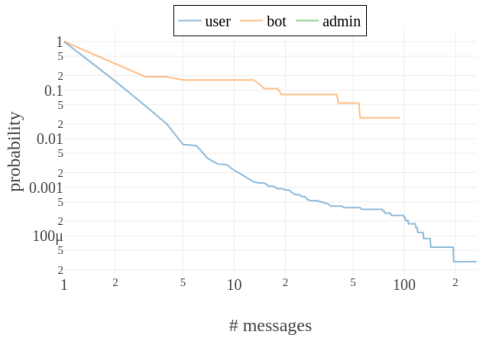
(l) gl



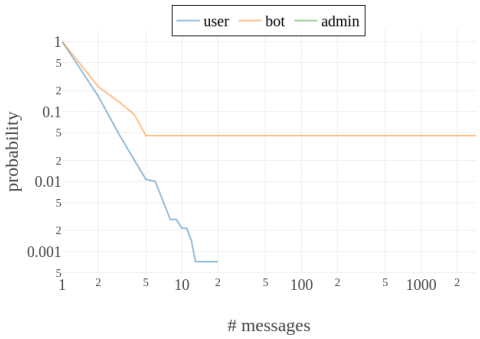
(m) ht



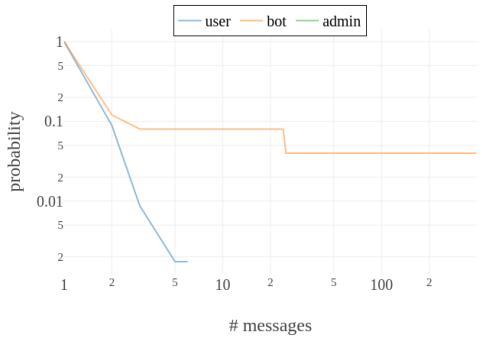
(n) it



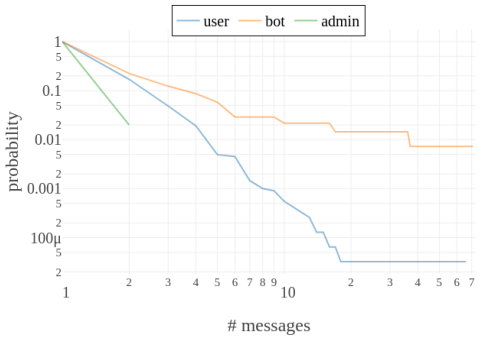
(o) ja



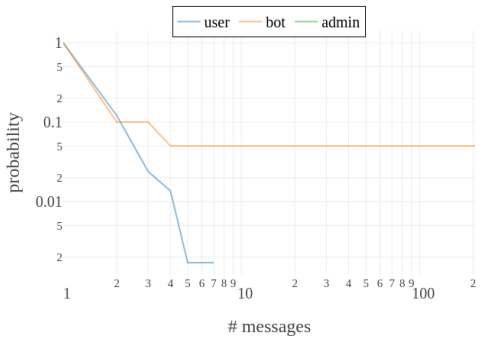
(p) lv



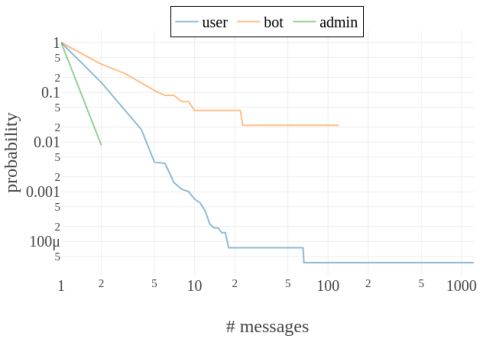
(q) nds



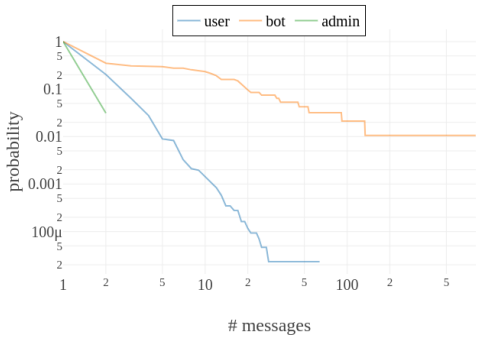
(r) nl



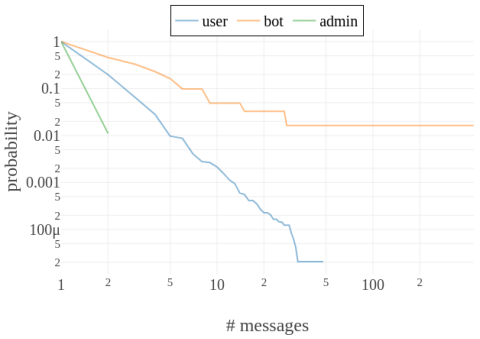
(s) oc



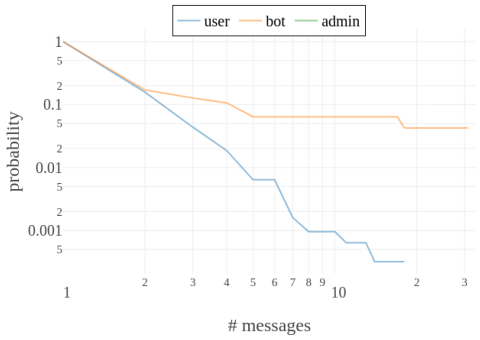
(t) pl



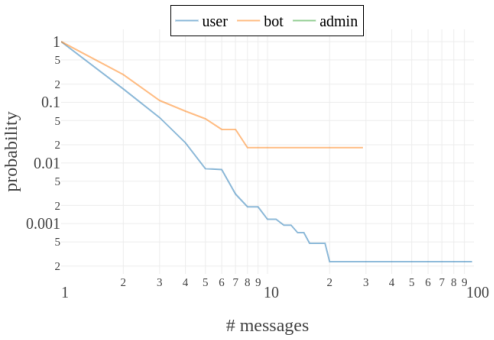
(u) pt



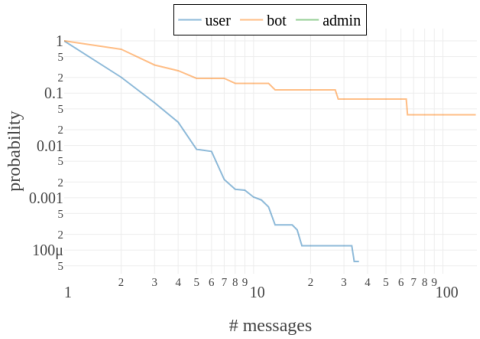
(v) ru



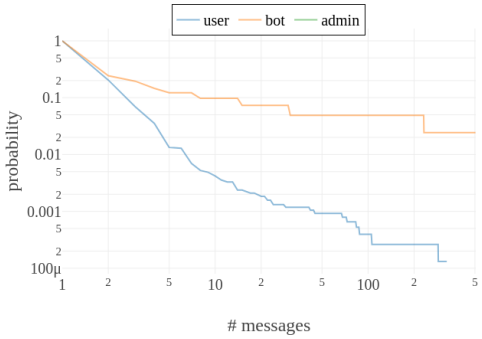
(w) sk



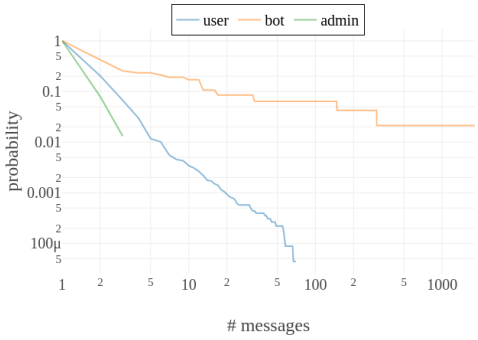
(x) sr



(y) sv



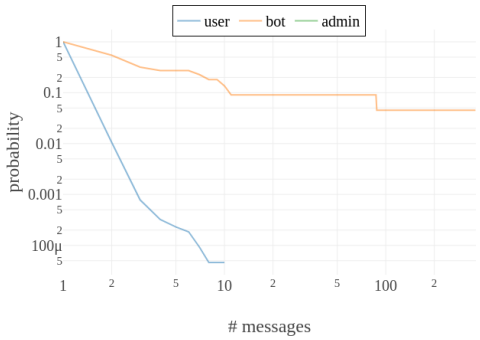
(z) vi



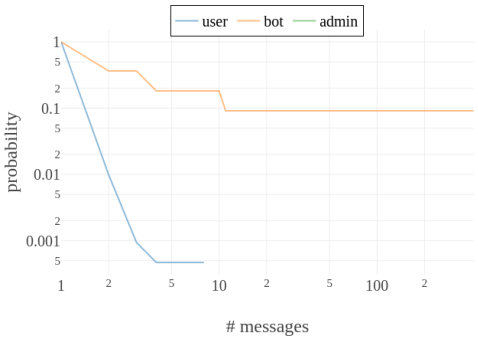
(aa) zh

Figure 44: Geometric mean of messages per hour.

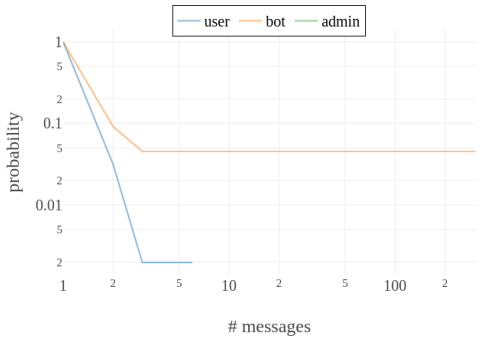
Geometric mean of messages per minute



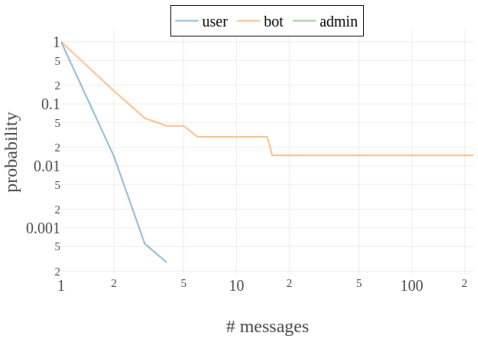
(a) ar



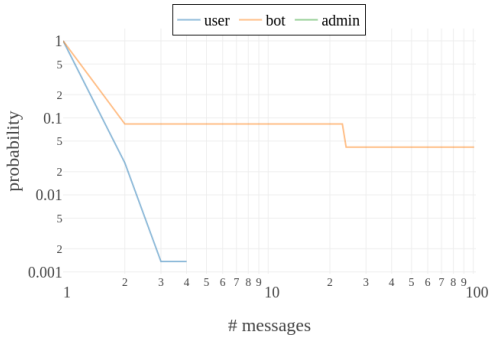
(b) bn



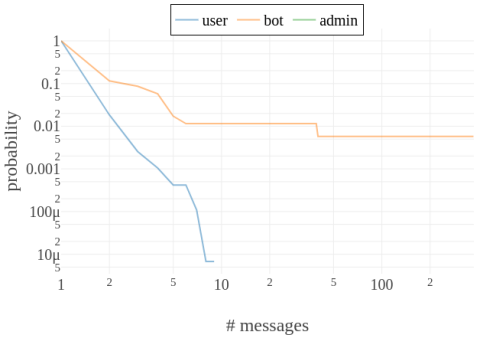
(c) br



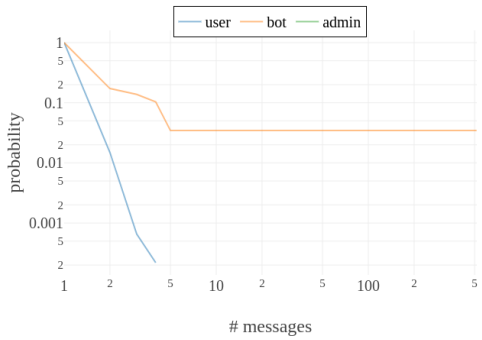
(d) ca



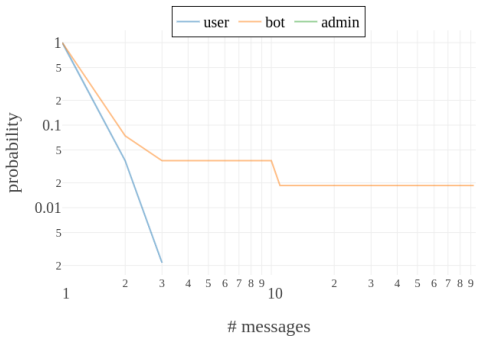
(e) cy



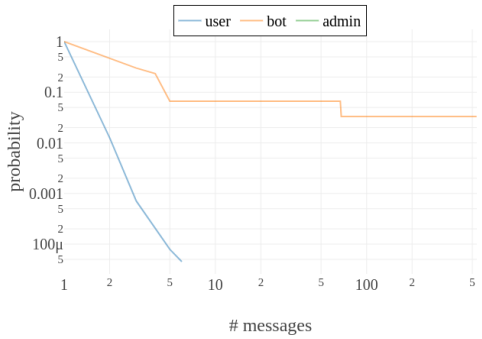
(f) de



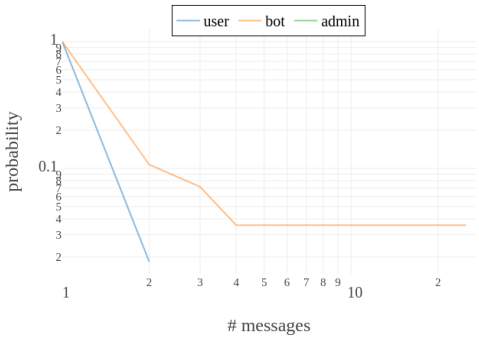
(g) el



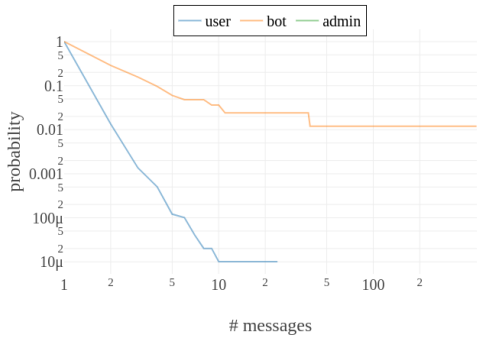
(h) eo



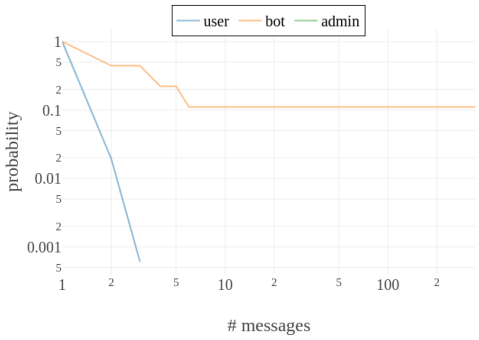
(i) es



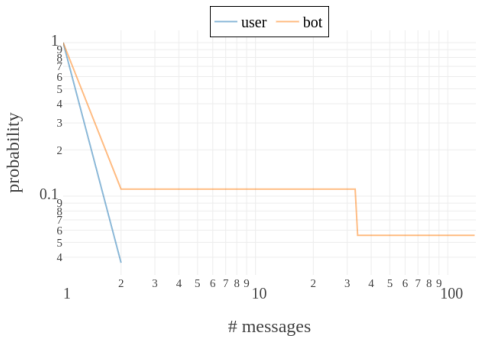
(j) eu



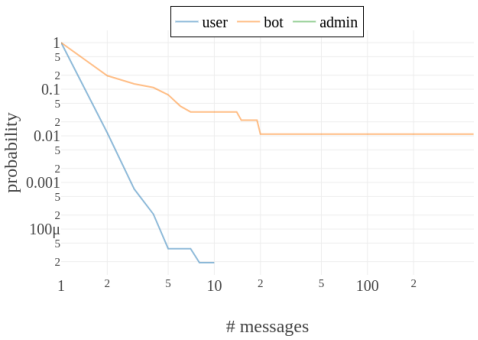
(k) fr



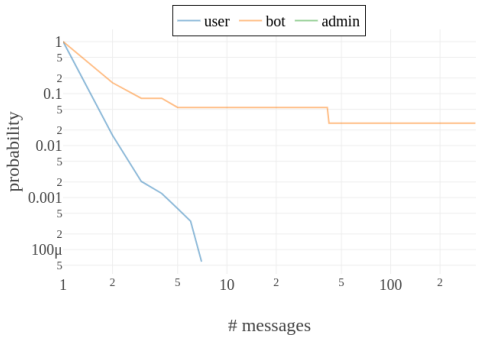
(l) gl



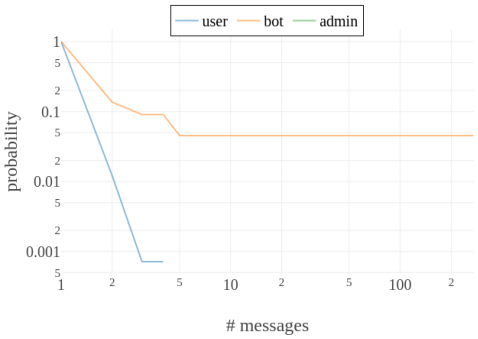
(m) ht



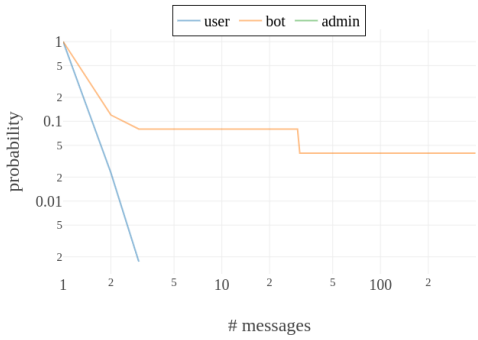
(n) it



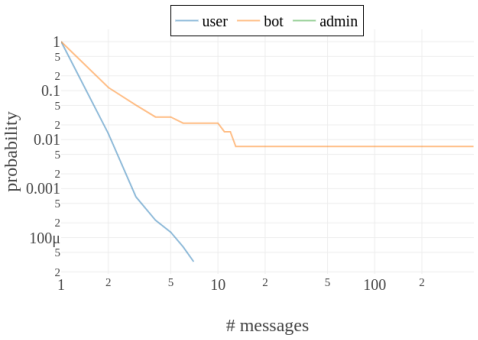
(o) ja



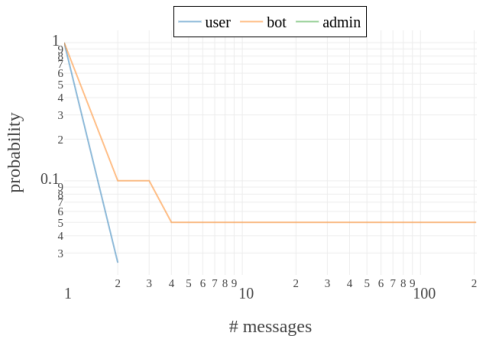
(p) lv



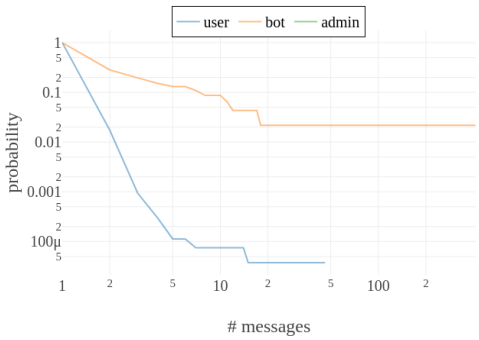
(q) nds



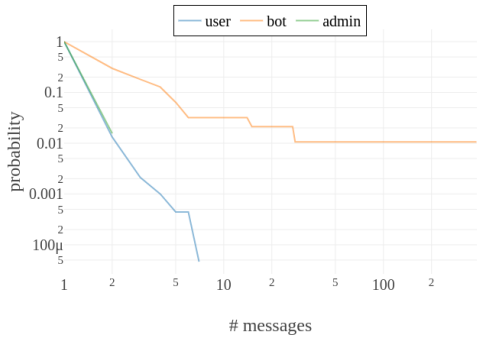
(r) nl



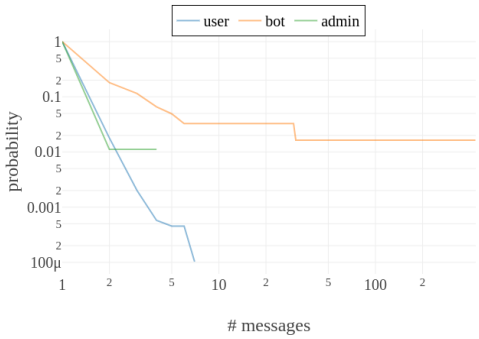
(s) oc



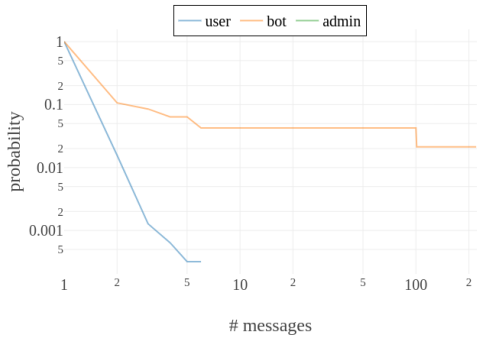
(t) pl



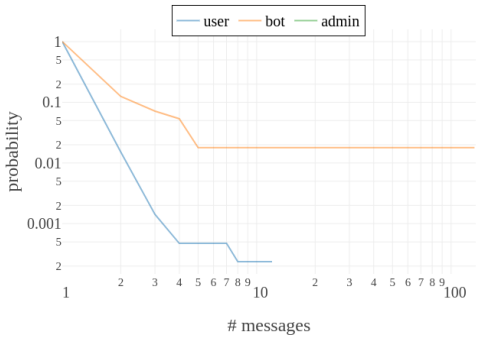
(u) pt



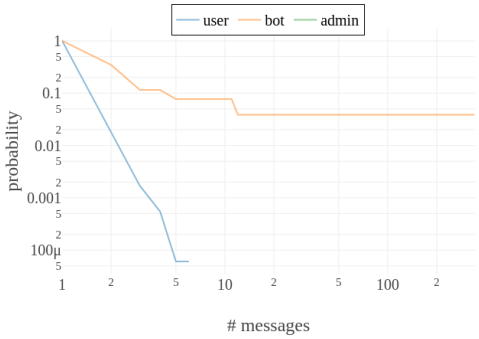
(v) ru



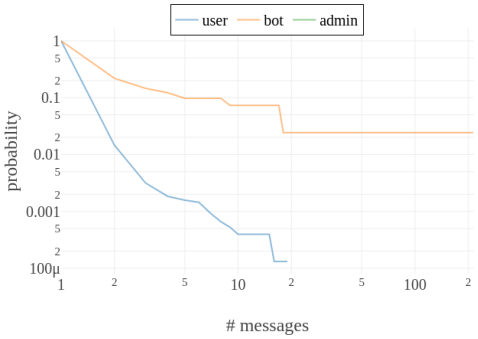
(w) sk



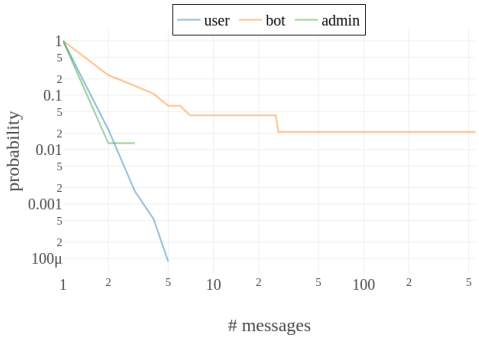
(x) sr



(y) sv



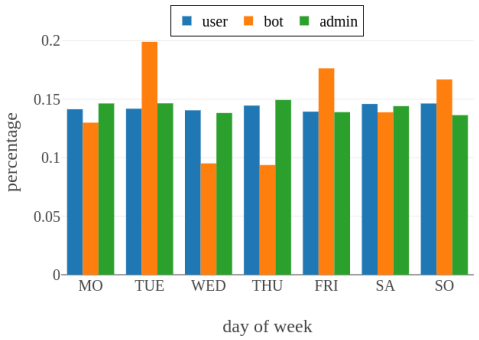
(z) vi



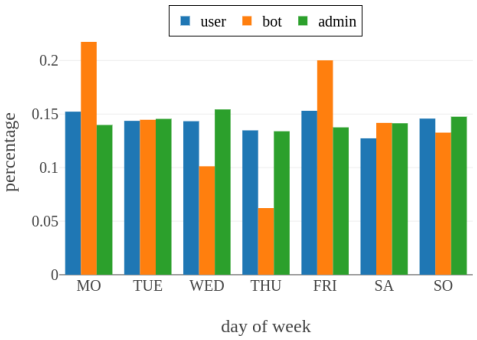
(aa) zh

Figure 45: Geometric mean of messages per minute.

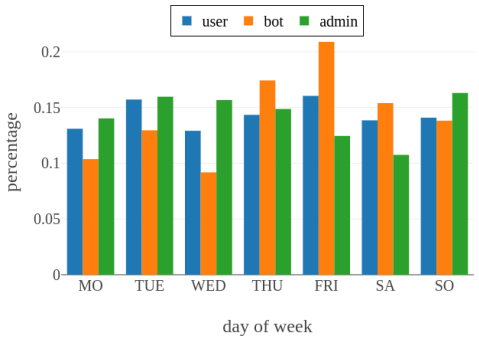
Weekday message activity



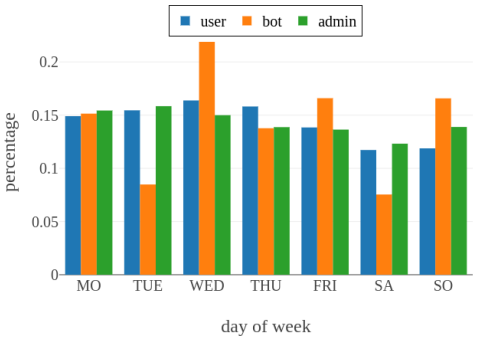
(a) ar



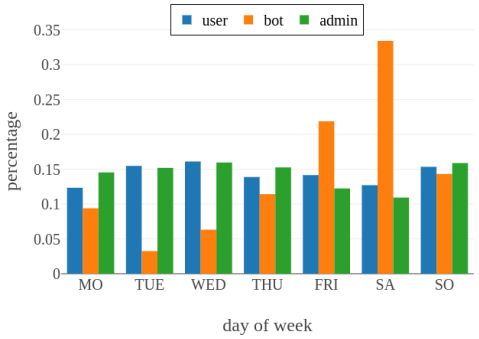
(b) bn



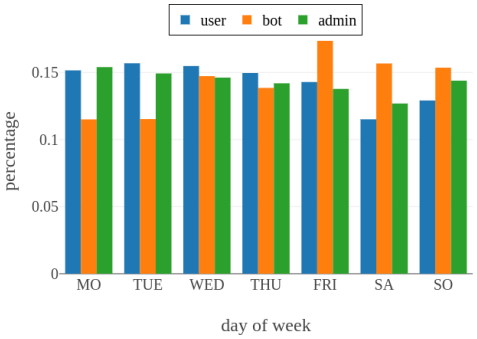
(c) br



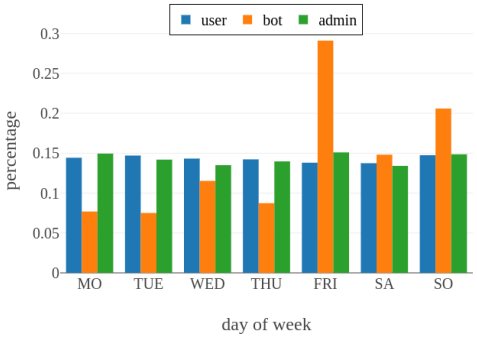
(d) ca



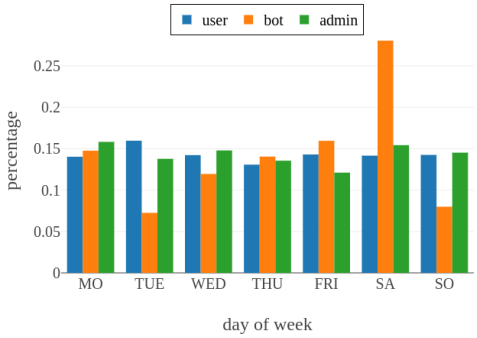
(e) cy



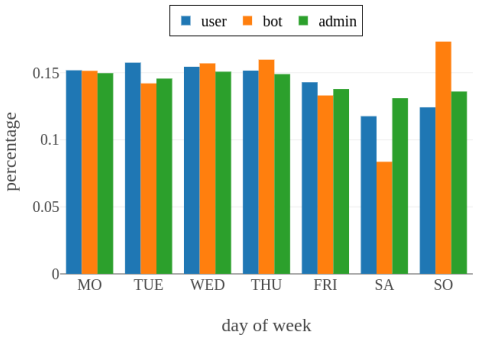
(f) de



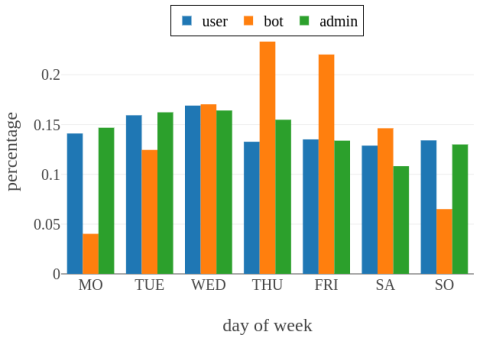
(g) el



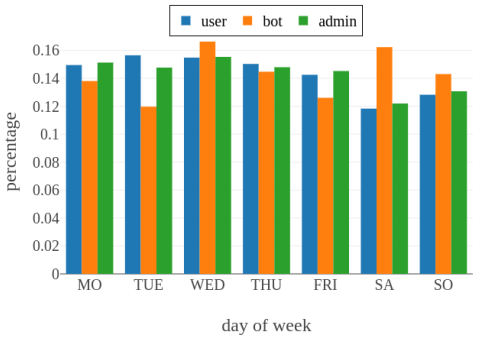
(h) eo



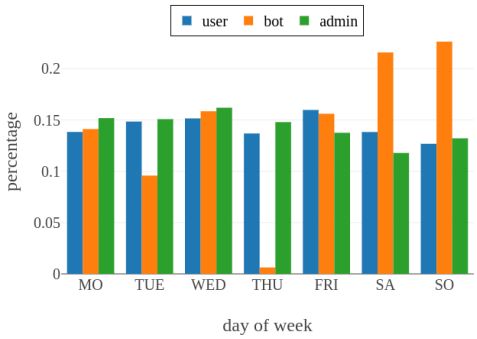
(i) es



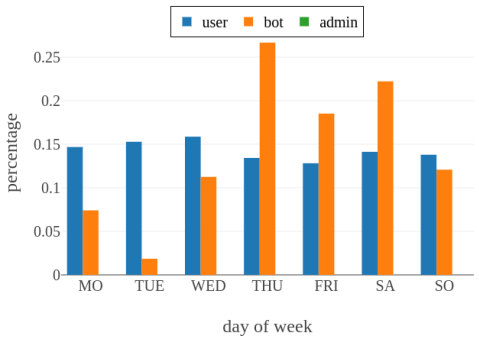
(j) eu



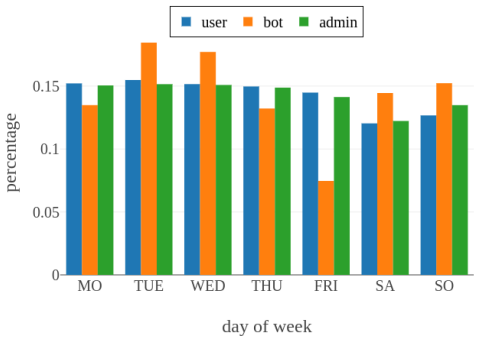
(k) fr



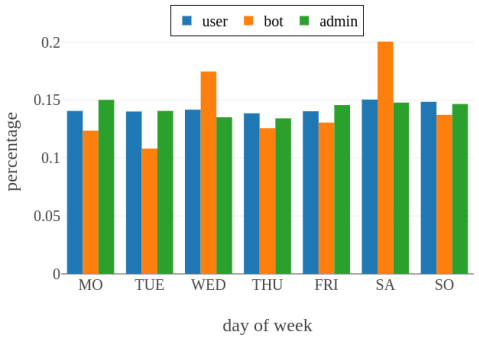
(l) gl



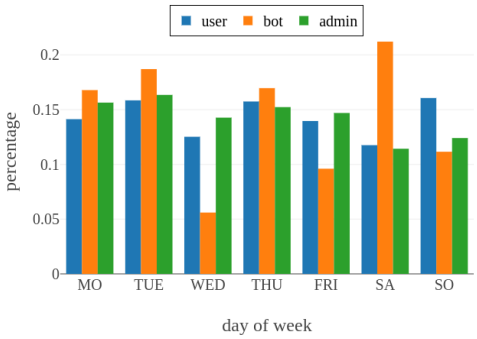
(m) ht



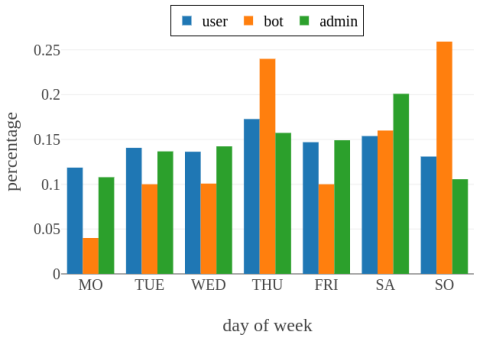
(n) it



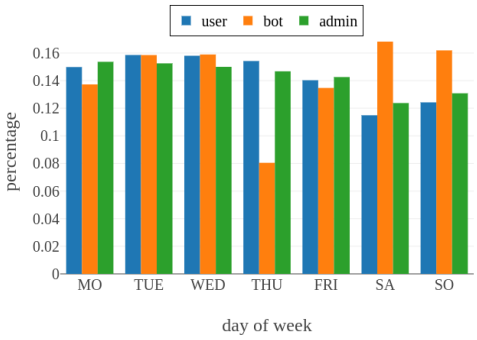
(o) ja



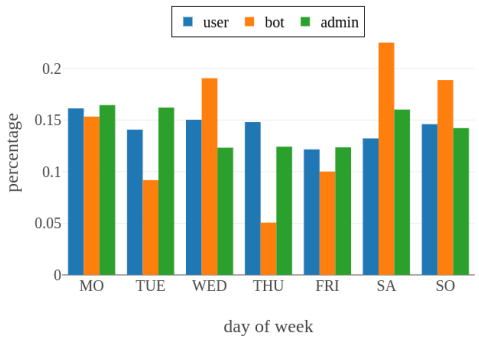
(p) lv



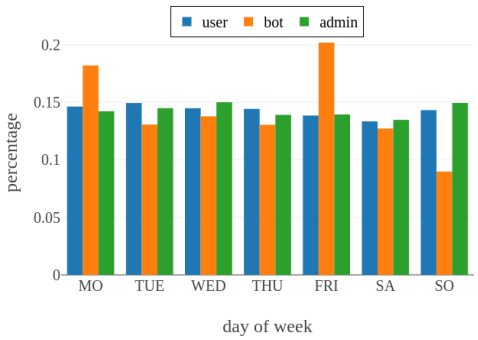
(q) nds



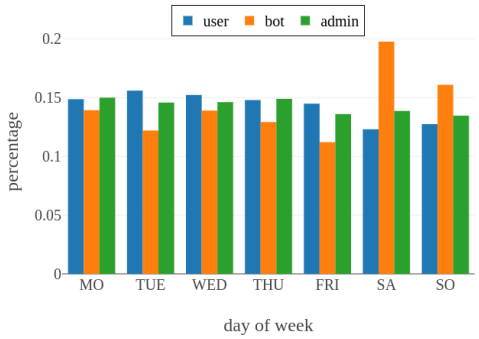
(r) nl



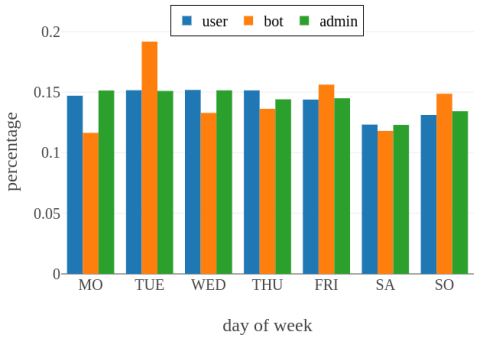
(s) oc



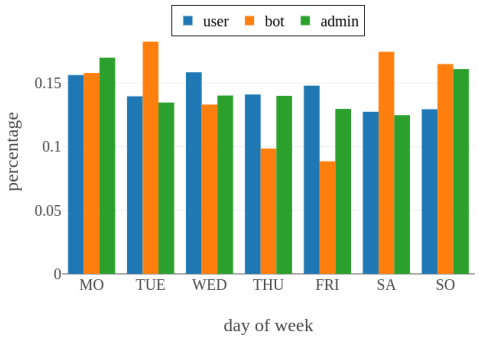
(t) pl



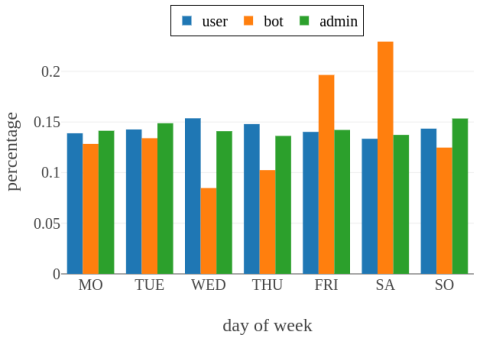
(u) pt



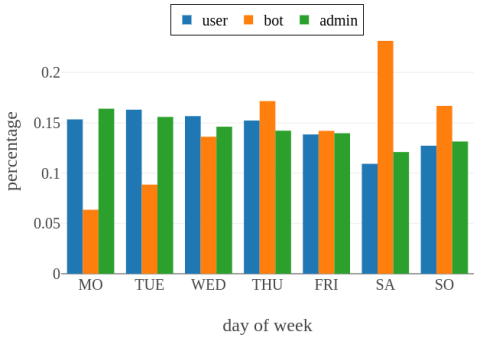
(v) ru



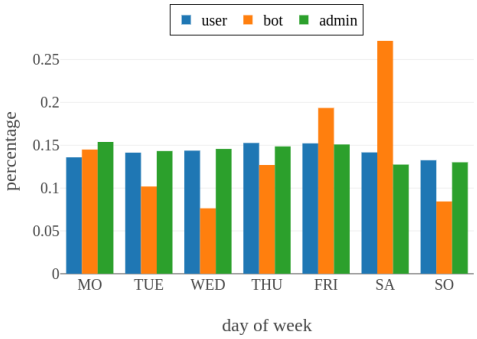
(w) sk



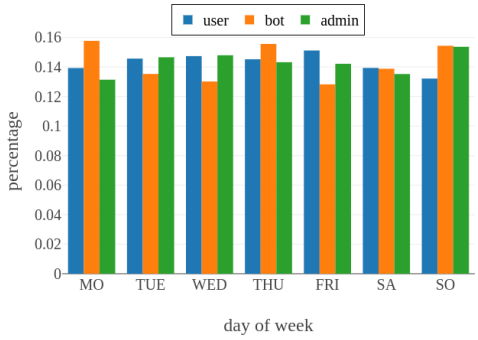
(x) sr



(y) sv



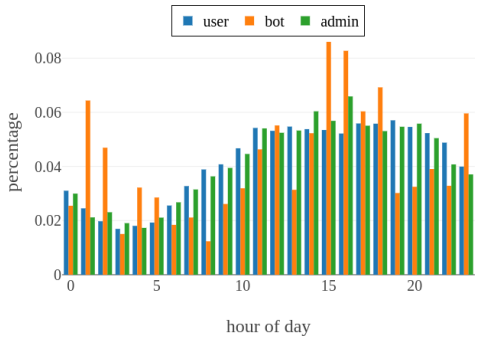
(z) vi



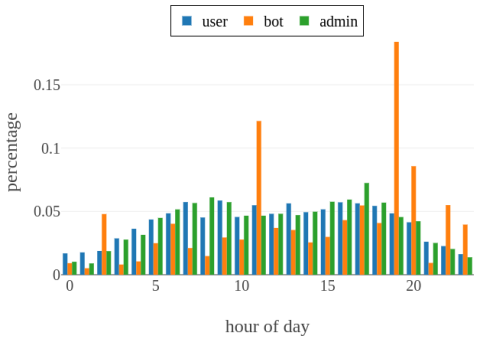
(aa) zh

Figure 46: Weekday message distribution by role.

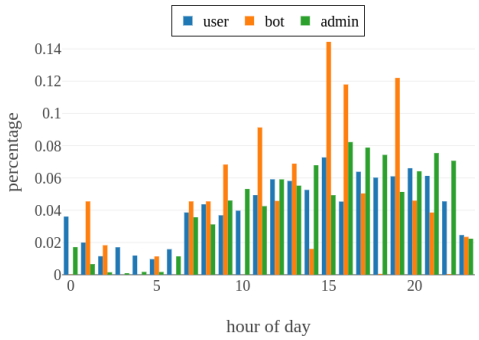
Day message activity



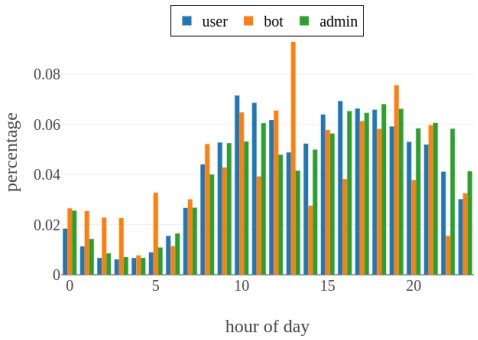
(a) ar



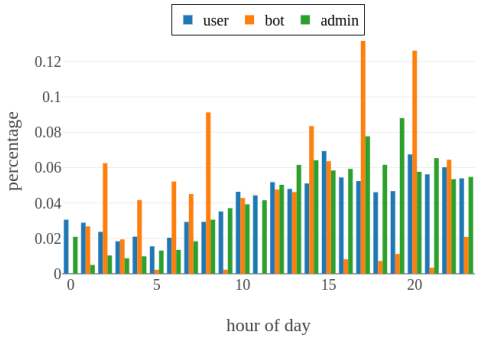
(b) bn



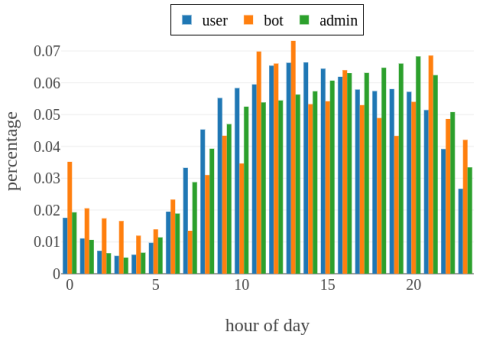
(c) br



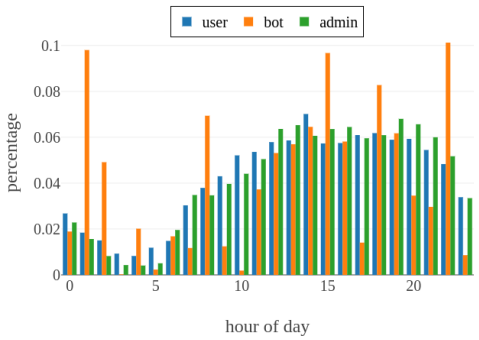
(d) ca



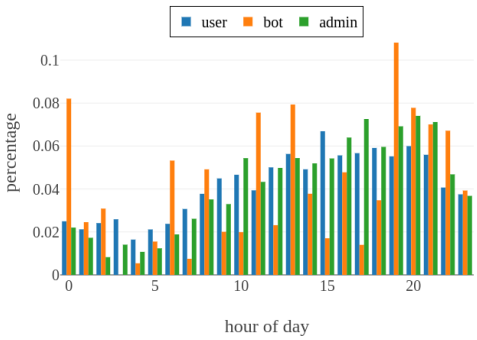
(e) cy



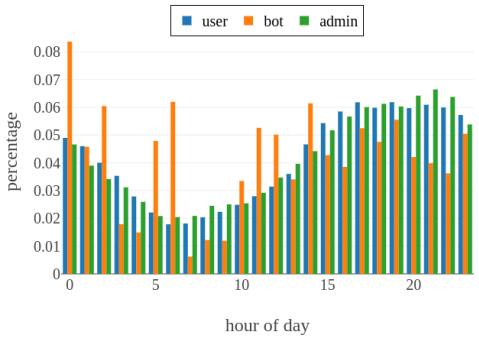
(f) de



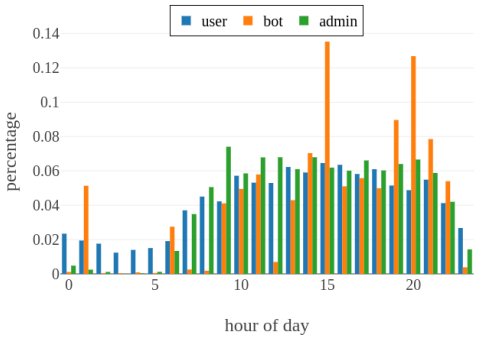
(g) el



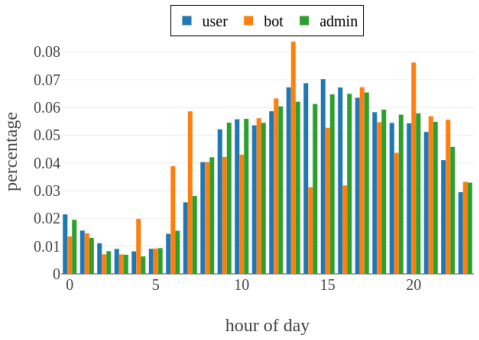
(h) eo



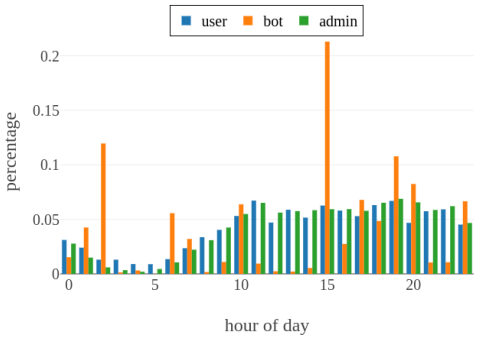
(i) es



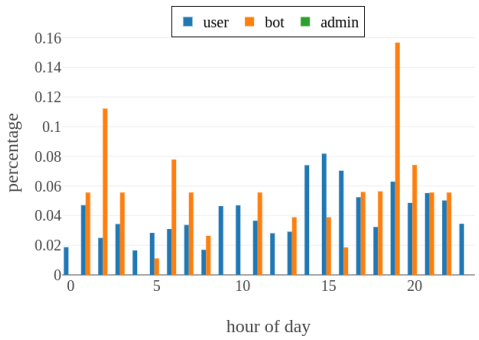
(j) eu



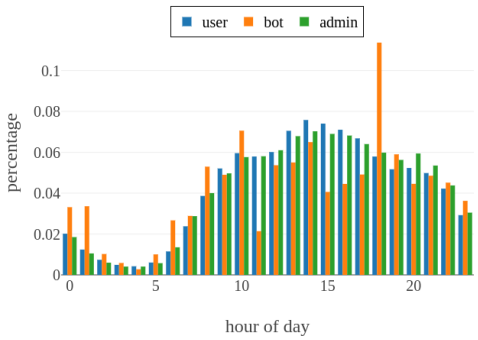
(k) fr



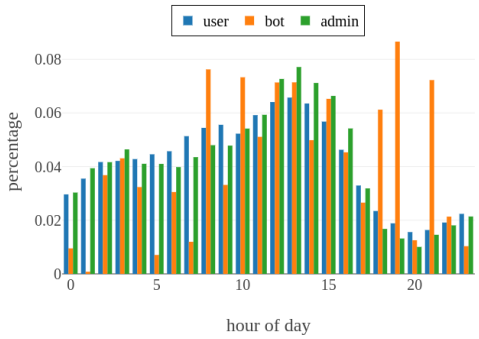
(l) gl



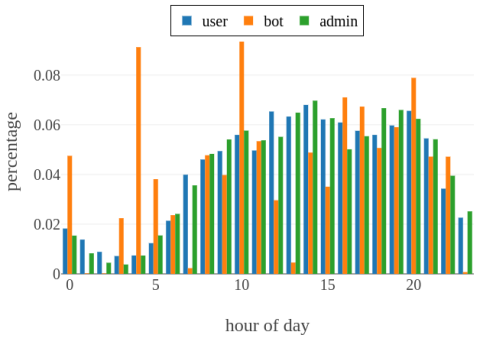
(m) ht



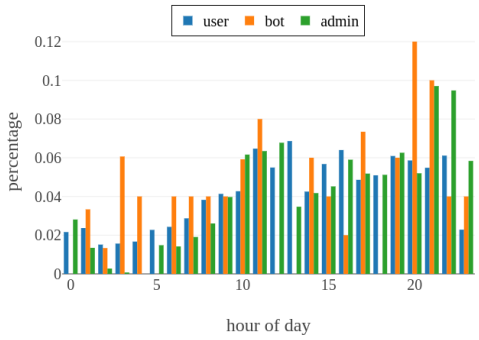
(n) it



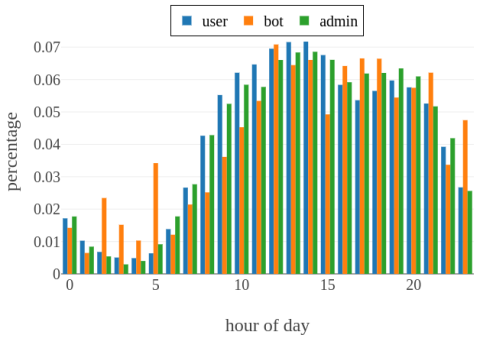
(o) ja



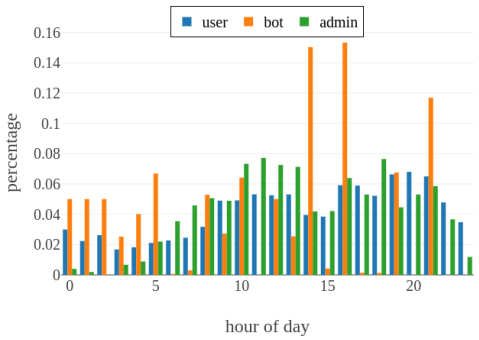
(p) lv



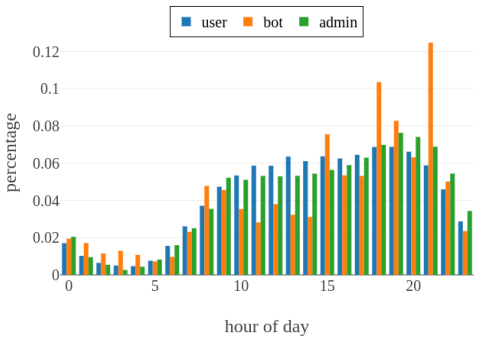
(q) nds



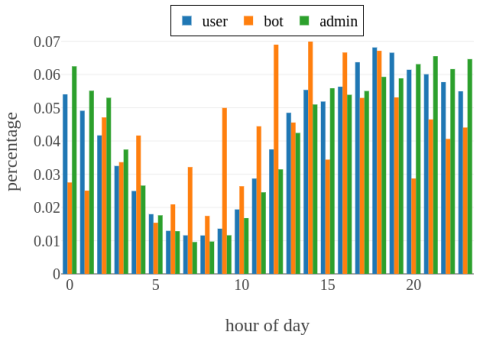
(r) nl



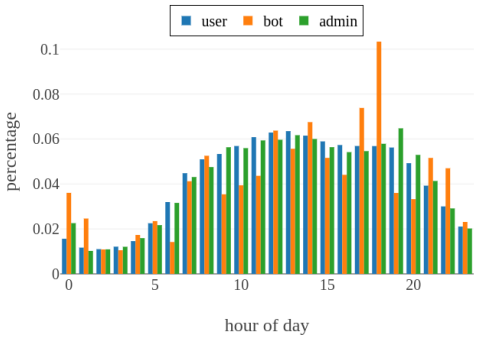
(s) oc



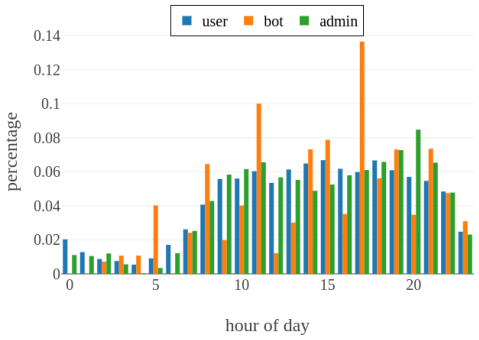
(t) pl



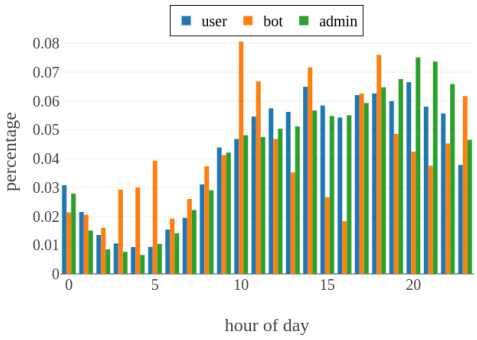
(u) pt



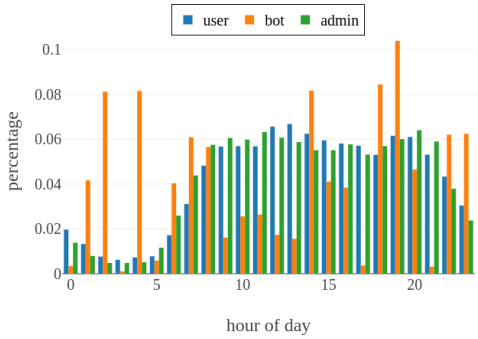
(v) ru



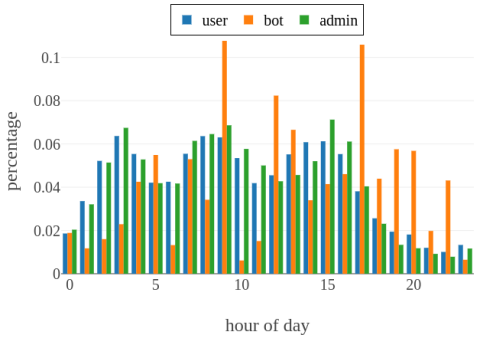
(w) sk



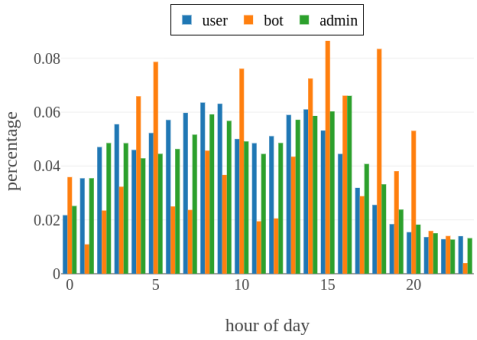
(x) sr



(y) sv



(z) vi



(aa) zh

Figure 47: Daily message distribution by role.

ROC-AUC scores

Baseline scores

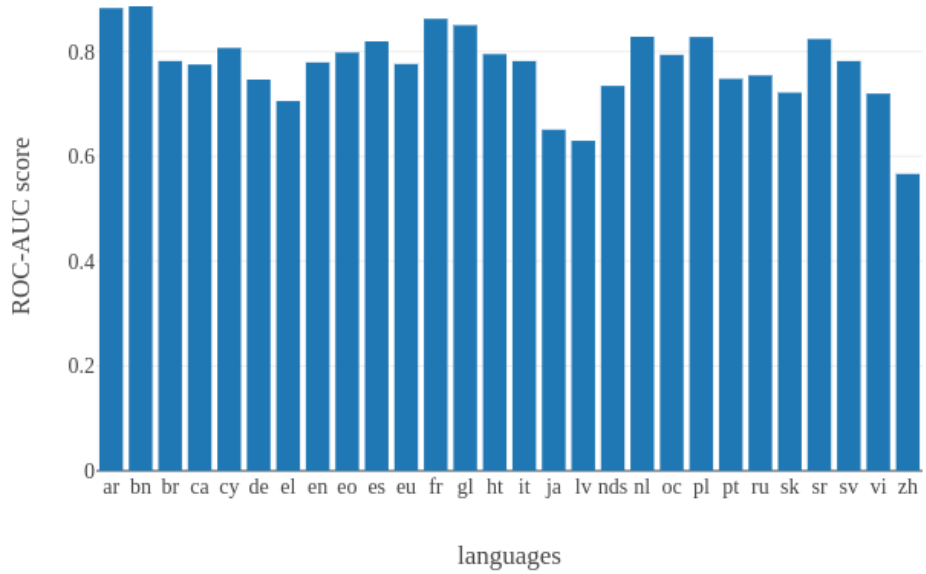


Figure 48: ROC-AUC scores of predicting bots for each sub datasets with the feature set of degree, indegree, outdegree, local clustering coefficient and PageRank. Used was a 10-fold cross validation within the given dataset.

Language	Score
ar	0.8829
bn	0.8865
br	0.782
ca	0.7752
cy	0.8068
de	0.7465
el	0.7056
en	0.7792
eo	0.7981
es	0.8195
eu	0.7763
fr	0.8625
gl	0.8502
ht	0.7951
it	0.7819
ja	0.6509
lv	0.6297
nds	0.7344
nl	0.8283
oc	0.7938
pl	0.8278
pt	0.748
ru	0.7544
sk	0.7216
sr	0.824
sv	0.7819
vi	0.7198
zh	0.5665

Table 9: ROC-AUC scores from the 10-Fold Cross Validation from figure 48.

Single enriched feature results

Results for individually added features to the baseline feature set of degree, in- and outdegree, PageRank and Clustering Coefficient. Value one within a column is the overall reached ROC-AUC score. Second value within a column is the difference between score of column one and the baseline scores.

Language	Activity Time	Weekday Entropy	Day Entropy	Coreness
ar	0.8828, -0.0001	0.883, 0.0001	0.8828, -0.0	0.797, 0.0019
bn	0.8607, -0.0258	0.8362, -0.0503	0.8804, -0.0061	0.7956, -0.0113
br	0.7946, 0.0126	0.7848, 0.0028	0.7827, 0.0007	0.7911, -0.0026
ca	0.7822, 0.007	0.7609, -0.0143	0.7652, -0.01	0.7879, 0.0059
cy	0.8366, 0.0297	0.821, 0.0141	0.8071, 0.0002	0.7162, -0.0182
de	0.7358, -0.0107	0.7363, -0.0102	0.7475, 0.001	0.8062, 0.0081
el	0.7211, 0.0155	0.7175, 0.0119	0.7184, 0.0129	0.802, 0.0257
eo	0.8001, 0.002	0.7958, -0.0024	0.7903, -0.0079	0.7862, -0.0641
es	0.8319, 0.0125	0.832, 0.0125	0.8194, -0.0	0.6448, 0.0151
eu	0.8133, 0.037	0.8012, 0.0249	0.8013, 0.025	0.8366, -0.0499
fr	0.8577, -0.0049	0.8254, -0.0371	0.8406, -0.0219	0.7391, 0.0175
gl	0.7184, -0.1319	0.784, -0.0662	0.7857, -0.0645	0.7014, -0.0042
ht	0.8083, 0.0132	0.7948, -0.0003	0.7998, 0.0047	0.8361, 0.0122
it	0.7859, 0.004	0.7554, -0.0265	0.778, -0.0039	0.7774, 0.0021
ja	0.6587, 0.0078	0.6391, -0.0118	0.6341, -0.0168	0.7677, -0.0142
lv	0.6498, 0.0202	0.625, -0.0046	0.6249, -0.0048	0.7212, 0.0015
nds	0.7473, 0.0129	0.7348, 0.0004	0.7508, 0.0164	0.6913, 0.0404
nl	0.8222, -0.0061	0.8101, -0.0182	0.8167, -0.0116	0.826, -0.0018
oc	0.8052, 0.0114	0.799, 0.0053	0.798, 0.0042	0.8168, -0.0115
pl	0.8361, 0.0083	0.8342, 0.0064	0.8528, 0.025	0.8828, -0.0001
pt	0.729, -0.019	0.7287, -0.0193	0.7198, -0.0282	0.7355, -0.0189
ru	0.724, -0.0304	0.7456, -0.0088	0.7612, 0.0068	0.6042, 0.0377
sk	0.7267, 0.005	0.7267, 0.0051	0.7472, 0.0256	0.7359, -0.0121
sr	0.8318, 0.0078	0.8194, -0.0046	0.8199, -0.0041	0.8029, -0.0166
sv	0.7834, 0.0014	0.8096, 0.0277	0.7838, 0.0018	0.7898, 0.0079
vi	0.7235, 0.0038	0.7572, 0.0375	0.7626, 0.0429	0.8513, -0.0112
zh	0.6043, 0.0378	0.587, 0.0205	0.5777, 0.0111	0.7477, 0.0012

Table 10: ROC-AUC scores for evaluation other features.

language	GM msg. # d.	GM msg. # h.	GM msg. # m.	AM msg. # d.	AM msg. # h.	AM msg. # m.
ar	0.8497, -0.0332	0.8497, -0.0332	0.8497, -0.0332	0.8497, -0.0332	0.8664, -0.0165	0.8831, 0.0002
bn	0.8608, -0.0257	0.8606, -0.0259	0.8611, -0.0254	0.8611, -0.0254	0.861, -0.0255	0.861, -0.0255
br	0.792, 0.01	0.7882, 0.0062	0.79, 0.008	0.7912, 0.0092	0.7862, 0.0042	0.7864, 0.0044
ca	0.7675, -0.0078	0.761, -0.0142	0.7661, -0.0091	0.7673, -0.0079	0.7731, -0.0022	0.7693, -0.0059
cy	0.8257, 0.0189	0.826, 0.0192	0.8239, 0.017	0.8239, 0.0171	0.8358, 0.029	0.8359, 0.0291
de	0.7416, -0.0049	0.7418, -0.0047	0.7463, -0.0002	0.7485, 0.002	0.7507, 0.0043	0.7571, 0.0106
el	0.7029, -0.0027	0.7181, 0.0125	0.7027, -0.0028	0.707, 0.0014	0.7114, 0.0058	0.7124, 0.0068
eo	0.8, 0.0019	0.8001, 0.002	0.7901, -0.008	0.7918, -0.0063	0.7976, -0.0006	0.7971, -0.001
es	0.8028, -0.0167	0.8154, -0.004	0.8152, -0.0042	0.8154, -0.0041	0.849, 0.0295	0.8323, 0.0128
eu	0.8021, 0.0259	0.8022, 0.0259	0.802, 0.0257	0.802, 0.0257	0.801, 0.0248	0.8011, 0.0248
fr	0.8295, -0.033	0.8372, -0.0254	0.8309, -0.0316	0.837, -0.0255	0.8376, -0.0249	0.8418, -0.0207
gl	0.8537, 0.0035	0.7221, -0.1282	0.7881, -0.0621	0.722, -0.1283	0.722, -0.1283	0.722, -0.1283
ht	0.7995, 0.0044	0.7975, 0.0024	0.8016, 0.0065	0.7983, 0.0032	0.7987, 0.0036	0.798, 0.0029
it	0.7708, -0.0111	0.7685, -0.0134	0.7897, 0.0078	0.7818, -0.0001	0.7925, 0.0106	0.7966, 0.0147
ja	0.6493, -0.0016	0.667, 0.0161	0.6512, 0.0003	0.6131, -0.0378	0.6259, -0.025	0.6675, 0.0166
lv	0.6413, 0.0117	0.6067, -0.023	0.6463, 0.0166	0.646, 0.0164	0.607, -0.0226	0.6069, -0.0228
nds	0.7673, 0.0329	0.7673, 0.0329	0.7673, 0.0329	0.7673, 0.0329	0.7673, 0.0329	0.7673, 0.0329
nl	0.8096, -0.0186	0.8216, -0.0066	0.8271, -0.0011	0.822, -0.0062	0.8115, -0.0168	0.8154, -0.0129
oc	0.7974, 0.0036	0.7971, 0.0033	0.7973, 0.0036	0.7972, 0.0034	0.7937, -0.0	0.7937, -0.0
pl	0.8529, 0.0251	0.853, 0.0252	0.8442, 0.0164	0.8263, -0.0015	0.8353, 0.0075	0.855, 0.0272
pt	0.7363, -0.0117	0.7332, -0.0148	0.7144, -0.0336	0.729, -0.019	0.732, -0.016	0.7467, -0.0013
ru	0.7242, -0.0303	0.7499, -0.0045	0.7616, 0.0072	0.7436, -0.0108	0.779, 0.0246	0.7658, 0.0114
sk	0.7335, 0.0119	0.7491, 0.0275	0.7409, 0.0192	0.7352, 0.0136	0.7279, 0.0062	0.7291, 0.0074
sr	0.8247, 0.0008	0.8213, -0.0026	0.8255, 0.0016	0.8297, 0.0057	0.8293, 0.0054	0.837, 0.013
sv	0.808, 0.0261	0.7826, 0.0007	0.7966, 0.0147	0.7974, 0.0155	0.7844, 0.0025	0.8149, 0.0329
vi	0.7576, 0.0379	0.7592, 0.0395	0.7436, 0.0238	0.7574, 0.0377	0.7487, 0.029	0.7496, 0.0298
zh	0.5859, 0.0194	0.6176, 0.0511	0.6174, 0.0509	0.5895, 0.023	0.627, 0.0605	0.6269, 0.0604

Table 11: ROC-AUC scores for evaluation with arithmetic interval and geometric message rates per time frame, where *d* are daily, *h* are hourly and *m* are minutely message rates.

Language	**AM interval**	**GM interval**	AM MIN	GM MIN	AM MAX	GM MAX
ar	0.8997, 0.0168	0.9164, 0.0335	0.8997, 0.0168	0.8997, 0.0168	0.8995, 0.0166	0.8995, 0.0166
bn	0.8553, -0.0312	0.8553, -0.0312	0.8553, -0.0312	0.8553, -0.0312	0.8613, -0.0252	0.8613, -0.0252
br	0.7828, 0.0008	0.7825, 0.0005	0.7823, 0.0003	0.7823, 0.0003	0.7825, 0.0005	0.7825, 0.0005
ca	0.765, -0.0102	0.7667, -0.0085	0.7655, -0.0098	0.7671, -0.0081	0.7616, -0.0136	0.7659, -0.0093
cy	0.8385, 0.0317	0.8387, 0.0319	0.8385, 0.0317	0.8385, 0.0317	0.8385, 0.0317	0.8385, 0.0317
de	0.7458, -0.0006	0.7407, -0.0058	0.7406, -0.0059	0.743, -0.0035	0.7469, 0.0004	0.7555, 0.009
el	0.7311, 0.0255	0.7316, 0.026	0.7159, 0.0104	0.716, 0.0104	0.7299, 0.0244	0.73, 0.0244
eo	0.7991, 0.001	0.8, 0.0018	0.8052, 0.0071	0.8, 0.0019	0.7993, 0.0011	0.8, 0.0018
es	0.8154, -0.0041	0.8488, 0.0294	0.8321, 0.0126	0.8028, -0.0167	0.8028, -0.0166	0.8028, -0.0166
eu	0.8022, 0.0259	0.8021, 0.0259	0.8022, 0.0259	0.8022, 0.0259	0.8022, 0.0259	0.8022, 0.0259
fr	0.8309, -0.0317	0.831, -0.0315	0.8158, -0.0467	0.8264, -0.0361	0.8355, -0.027	0.8463, -0.0162
gl	0.7847, -0.0655	0.7196, -0.1306	0.7191, -0.1311	0.7193, -0.131	0.786, -0.0642	0.7845, -0.0657
ht	0.7922, -0.0029	0.7915, -0.0036	0.7876, -0.0075	0.7909, -0.0042	0.7902, -0.0049	0.7922, -0.0029
it	0.7605, -0.0214	0.7767, -0.0052	0.7568, -0.0251	0.7766, -0.0053	0.7666, -0.0153	0.7728, -0.0091
ja	0.64, -0.0109	0.6608, 0.0099	0.6582, 0.0073	0.6688, 0.0179	0.6403, -0.0106	0.6402, -0.0107
lv	0.6304, 0.0007	0.6304, 0.0007	0.6304, 0.0007	0.6304, 0.0007	0.6304, 0.0007	0.6304, 0.0007
nds	0.7837, 0.0493	0.7837, 0.0493	0.7837, 0.0493	0.7837, 0.0493	0.7837, 0.0493	0.7837, 0.0493
nl	0.8048, -0.0235	0.808, -0.0202	0.8105, -0.0178	0.8145, -0.0138	0.7974, -0.0309	0.8057, -0.0226
oc	0.8024, 0.0087	0.8024, 0.0086	0.8024, 0.0086	0.8024, 0.0086	0.8026, 0.0088	0.8026, 0.0088
pl	0.8342, 0.0064	0.8444, 0.0166	0.844, 0.0162	0.844, 0.0162	0.8358, 0.008	0.8359, 0.0081
pt	0.7388, -0.0092	0.7362, -0.0118	0.741, -0.007	0.7349, -0.0131	0.7384, -0.0096	0.7356, -0.0124
ru	0.7529, -0.0015	0.7211, -0.0333	0.7131, -0.0413	0.7107, -0.0437	0.7423, -0.0121	0.733, -0.0214
sk	0.7365, 0.0149	0.7355, 0.0139	0.7262, 0.0045	0.7265, 0.0048	0.7284, 0.0067	0.7286, 0.007
sr	0.8281, 0.0041	0.8336, 0.0096	0.8357, 0.0118	0.8315, 0.0075	0.8295, 0.0055	0.8257, 0.0017
sv	0.8098, 0.0279	0.8096, 0.0276	0.8091, 0.0272	0.8089, 0.027	0.8098, 0.0278	0.8098, 0.0279
vi	0.7353, 0.0156	0.7353, 0.0156	0.7353, 0.0155	0.7353, 0.0155	0.7287, 0.0089	0.7286, 0.0089
zh	0.6213, 0.0548	0.6529, 0.0864	0.6214, 0.0548	0.6316, 0.0651	0.6205, 0.054	0.6208, 0.0542

Table 12: ROC-AUC scores for evaluation with arithmetic and geometric interval as well as their minimum and maximum.