Conversational Networks for Automatic Online Moderation

Etienne Papegnies, Vincent Labatut, Richard Dufour, and Georges Linarès

***Abstract*— Moderation of user-generated content in an online community is a challenge that has great socio-economic ramifications. However, the costs incurred by delegating this paper to human agents are high. For this reason, an automatic system able to detect abuse in user-generated content is of great interest. There are a number of ways to tackle this problem, but the most commonly seen in practice are word filtering or regular expression matching. The main limitations are their vulnerability to intentional obfuscation on the part of the users, and their context-insensitive nature. Moreover, they are language dependent and may require appropriate corpora for training. In this paper, we propose a system for automatic abuse detection that completely disregards message content. We first extract a conversational network from raw chat logs and characterize it through topological measures. We then use these as features to train a classifier on our abuse detection task. We thoroughly assess our system on a dataset of user comments originating from a French massively multiplayer online game. We identify the most appropriate network extraction parameters and discuss the discriminative power of our features, relatively to their topological and temporal nature. Our method reaches an F-measure of 83.89 when using the full feature set, improving on existing approaches. With a selection of the most discriminative features, we dramatically cut computing time while retaining the most of the performance (82.65).**

***Index Terms*— Classification algorithms, Information retrieval, Network theory (graphs), Social computing, Text analysis.**

I. Introduction

ONLINE communities have acquired an indisputable importance in today's society. From modest beginnings as places to trade ideas around specific topics, they have grown into important focuses of attention for companies to advertize products or governments interested in monitoring public discourse. They also have a strong social effect, by heavily impacting public and interpersonal communications.

However, the Internet grants a degree of anonymity, and because of that, online communities are often confronted with users exhibiting abusive behaviors. The notion of abuse varies depending on the community, but there is almost always a common core of rules stating that users should not personally attack others or discriminate them based on race, religion, or sexual orientation. It can also include more community-specific aspects, e.g., not posting advertisement or external URLs. For community maintainers, it is often necessary to act on abusive behaviors: if they do not, abusive users can poison the community, make important community members leave, and, in some countries, trigger legal issues [1], [2].

When users break the community rules, sanctions can then be applied. This process, called moderation, is mainly done by humans. Since this manual work is expensive, companies have a vested interest in automating the process. In this paper, we consider the classification problem consisting in automatically determining if a user message is abusive or not. This task is at the core of automated moderation, and it is difficult for several reasons. First, the amount of noise in the content (typos, grammatical errors, uncommon abbreviations, out-of-vocabulary words...) of messages posted on the Internet is usually quite high. Furthermore, while some of this noise is unwittingly produced by fast typing or poor language skills, a good part of it is voluntarily introduced as a means to defeat automated badword checks, e.g. "Pls d1e you f8 ck". Then, even with a noiseless message, it is sometimes necessary to perform advanced natural language analysis to detect abuse in a message. Here is a fictional example of a message containing no obvious indicators of abuse such as straight insults, while still being very abusive indeed: "Would you like to meet your maker? I can arrange that". Finally, even advanced natural language processing approaches may not be able to detect abuse from a message only without looking at its context. This context can take various forms. For instance, in the case of a "Yo mama joke", it is the continuation of the conversation. But it can also include external knowledge, which makes it harder to handle. Consider the following exchange, for example: *A:* "They've been discriminated against enough. Six millions of them were killed during the holocaust." – *B:* "That didn't actually happen". The message from B has no abuse markers at all until one considers both the messages that came before and historical knowledge.

To address these issues, we propose, as our main contribution in this paper, an approach that completely ignores the content of the messages and models conversations under the form of conversational graphs. By doing so, we aim to create a model that is not vulnerable to text-based obfuscation. We characterize these graphs through a number of topological measures which are then used as features, in order to train and test a classifier. Our second contribution is to apply our method to a corpus of chat logs originating from the community of the French massively multiplayer online game SpaceOrigin.[1]

Manuscript received April 16, 2018; revised September 25, 2018; accepted December 6, 2018. Date of publication January 29, 2019; date of current version February 12, 2019. This work was supported in part by Provence-Alpes-Côte-d'Azur region, France and in part by Nectar de Code Company. *(Corresponding author: Etienne Papegnies.)*

E. Papegnies is with the Laboratoire Informatique d'Avignon, Avignon University, 84911 Avignon, France, and also with Nectar de Code, 13570 Barbentane, France (e-mail: etienne.papegnies@univ-avignon.fr).

V. Labatut, R. Dufour, and G. Linares are with the Laboratoire Informatique d'Avignon, Avignon University, 84911 Avignon, France.

Digital Object Identifier 10.1109/TCSS.2018.2887240

[1] https://play.spaceorigin.fr/

Our third contribution is to investigate the relative importance of the classification features, as well as the parameters of the graph extraction process, with regard to our classification task—the detection of abusive messages.

This paper is a significantly extended version of our preliminary work started in [3]. In comparison, we propose and experiment with several variations of our network extraction method and vastly expand the array of features that we consider. We also adapt our approach to greatly increase the efficiency of our system with regard to necessary computational resources and make it more versatile to possible use cases.

The rest of this paper is organized as follows. In Section II, we review related work on abuse detection and previous approaches dedicated to network extraction from various types of conversation logs. We describe the methods used throughout our pipeline in Section III, including the approach proposed to extract conversational networks, and the topological features that we compute to characterize them. In Section IV, we present our dataset, as well as the overall experimental setup for the classification task. We then provide a discussion and a qualitative study of the performance of our approach, with a focus on the contributions of the considered features. Because some of them are computed from information that is not yet available at the instant some messages are posted, we also examine the performances of the system-based only on information available at the time (i.e., as a prediction task). Finally, we summarize our contributions in Section V and present some perspectives for this paper.

II. Related Work

In this section, we first review general approaches related to the problem of abuse detection (Section II-A), and then focus on techniques that have been previously used to extract graph-based representations of conversation logs (Section II-B).

A. Abuse Detection

One can distinguish two main categories of works related to abuse detection: those using the content of the targeted messages only and those focusing on their context (user metadata, content of surrounding messages...). Some hybrid works also propose to combine both categories.

1) Content-Based Approaches: The work initiated by Spertus in [4] constitutes a first attempt to create a classifier for hostile messages. Abusive messages often contain hostility, so this task is related to ours. However, the notion of abuse is more general, as it can take a nonhostile form. Spertus uses static rules to extract linguistic markers for each message: imperative statement, profanity, condescension, insult, politeness, and praise. These are then used as features in a binary classifier. This approach obtains good results, except in specific cases like hostility through sarcasm. However, manually defining all the linguistic rules related to an abusive message is a severe limitation and appears impossible, in practice. Also, its application to another language would require to transpose it to other grammar rules and idioms.

Chen *et al.* [5] seek to detect offensive language in social media so that it can be filtered out to protect adolescents. Like before, this task is more specific than ours, as using offensive language is just one type of abuse. Chen *et al.* [5] developed a system that uses lexical and syntactical features as well as user modeling, to predict the offensiveness value of a comment. They note that the presence of a word tagged as offensive in a message is not a definite indication that the message itself is offensive. For instance, while "you are stupid" is clearly offensive, "this is stupid xD" is not. They further show that lack of context can be somewhat mitigated by looking at word n-grams instead of unigrams (i.e., single words). The method relies on manually constituted language-dependent resources though, such as a lexicon of offensive terms, which also makes it difficult to transpose to another language.

Dinakar *et al.* [6] use tf–idf features, a static list of badwords, and of widely used sentences containing verbal abuse, to detect cyberbullying in Youtube comments. Bullying is mainly characterized by its persistent and repetitive nature, and it can, therefore, be considered as a very specific type of abuse. Like before, the proposed model shows good results except when sarcasm is used. It is worth noting that sarcasm can be considered as a form of natural language obfuscation that is especially hard to detect in written communications, because of the lack of inflection clues.

Chavan and Shylaja [7] review machine learning (ML) approaches to detect cyberbullying in online social networks. They show that pronoun occurrences, usually neglected in text classification, are very important to detect online bullying. They use skip-gram features to mitigate the sentence-level context issues by taking into account distant words. These new features allow them to boost the accuracy of a classifier detecting bullying by 4% points. The approach is, however, still vulnerable to involuntary misspellings and word-level obfuscation. It uses a language-dependent list of badwords during preprocessing.

In their recent article, Mubarak *et al.* [8] work on the detection of offensive language in Arabic media, by introducing the interesting possibility of dynamically generating and expanding a list of bad words. They extract a corpus of tweets that is divided into two classes (obscene/not obscene) based on static rules. Then, they perform a log odds ratio analysis to detect the words favoring documents from the obscene class. Such an approach could be very useful in an online classification setting, but inherently requires a dataset where the number of samples in the obscene class is large. Still, they show that a list of words dynamically generated using that method contains 60% of new obscene words, and the process can be iterated over. Relatively to our problem of interest, the main limitation of this paper is its focus on obscene words, which are just one specific type of abuse.

Razavi *et al.* [9] focus on a wider spectrum of types of abuse than the previously cited works, which they call inflammatory comments. It ranges from impoliteness to insult, and includes rants and taunts. To detect them, they stack three levels of Naive Bayes classifier variants, fed with features related to the presence, frequency, and strength of offensive expressions. These are computed based on a manually constituted lexicon of offensive expressions and insults, which makes the method relatively corpus-specific. The resulting system shows high

precision and has the useful characteristics of being updatable online. It is, however, vulnerable to the text-based obfuscation techniques we have previously mentioned.

With recent developments in GPU architecture and hardware availability, more computationally expansive techniques have been used. Djuric *et al*. [10] detect hate speech in Twitter data. They adopt a two-step approach consisting in first learning a low-dimensional representation of the tweets, and then applying a classifier to discriminate them based on this representation. They note that jointly using message- and word embeddings instead of simple bag-of-words boosts the performance. Park and Fung [11] also work on tweets using neural networks, but they focus only on sexism- and racism-related cases. They propose a two-step framework consisting in first training a convolutional neural network (CNN) to identify the absence/presence of abuse, and then performing a simple logistic regression to further discriminate between sexism and racism. Both of these approaches are inherently portable, however, they require a lot of data.

Pavlopoulos *et al*. [12] develop an automatic moderation system for comments posted by users on websites. It is based on a recurrent neural network operating on word embeddings, with an attention mechanism. They apply it to two large corpus extracted from a Greek sports website and the English Wikipedia. The proposed system outperforms CNN and other more mainstream classifiers. However, it is worth noticing that these tasks are slightly different, as the the Greek corpus is annotated for general moderation, whereas the English one focuses on personal attacks.

It is worth noting that all these ML-based approaches perform better when a large dataset is available for training. However, text-based approaches are usually language dependent, which means that models have to be trained on a dataset of the specific language. This is usually not an issue when classifying English messages because of the wealth of publicly available data but is problematic in our case, since our messages come from low-resource language communities.

Content-based text classification usually makes for a good baseline. However, such methods have severe limitations. First, abuse can be spread over a succession of messages. Some messages can even reference a shared history between two users. Second, it is very common for users to voluntarily obfuscate message content to work around badwords detection. Indeed, abusers can bypass automatic systems by making the abusive content difficult to detect: for instance, they can intentionally modify the spelling of a forbidden word.

Hosseini *et al*. [13] demonstrate such an attack against the Google Perspective API.[2] Adversarial attacks based on word-level obfuscation are nothing new, and approaches exist to counter them. For instance, Lee and Ng [14] experiment with spam de-obfuscation using a hidden Markov model that incorporates lexical information. Such an approach yields good results for de-obfuscation, but it is computationally expensive and requires a dataset of obfuscated words for training. More recently, Rojas-Galeano [15] describes a more compact approach based on a dynamic programing sequence alignment algorithm. It has a different set of limitations, the main one being that it does not allow for one character to be used as an obfuscated version of several distinct original characters (it uses a one-to-one character mapping).

2) Context-Based Approaches: Because the reactions of other users to an abuse case are completely beyond the control of the abuser, some works consider the content of messages *around* the targeted message, instead of the content of the targeted message only.

For instance, Yin *et al*. [16] use features derived from the sentences neighboring a given message to detect harassment on the Web. Harassment implies repetition and can be considered as a specific type of abuse. Their goal is to spot conversations going off-topic and use that as an indicator. Their combined content/context approach shows good results when used against multiparticipant chat logs. They also note that sentiment features seem to constitute mostly noise due to the high misspelling rate. This lack of discriminative power from sentiment features is something we have also noticed while experimenting with content-based techniques on our data in [17].

Cheng *et al*. [18] do not try to perform automatic moderation. Instead, they conduct a comprehensive study of antisocial behavior in online discussion communities, and use its results to build user behavior models. We include this paper in our review, because it provides some insight into the devolution of abusive users over time in a community, regarding both the quality of their contributions and their reactions toward other members of the community. A critical result of this analysis is that instances of antisocial messages usually generate a bigger response from the community, compared to normal messages. In our own work, we build upon this observation and compare classification performances obtained when considering or ignoring messages published right after the classified message.

Balci and Salah [19] take advantage of user features to detect abuse in the community of an online game. These features include information such as gender, number of friends, financial investment, avatars, and general rankings. The goal is to help human moderators dealing with abuse reports, and the approach yields sufficiently good results to achieve this. One important difference with our work is that in our case, the user data necessary to replicate this approach are not available. As a practical consideration the availability of that data will always depend on the type of the community.

In [17], we tackle the same problem as in this paper, i.e., detect abuse in chat messages in the context of an online game. However, unlike the method proposed presently, we use a wide array of language features (bag-of-words, $tf\text{-}idf$ scores, sentiment scores…) as well as context features derived from the language models of other users. We also experiment with several advanced preprocessing approaches. This method allows us to reach a performance of 72.1% in terms of F-measure on our abusive message detection task.

Of all the approaches of the literature described in this section, [17] as well as Balci and Salah's [19] aim at solving the same problem as us. The others focus on tasks which are related to abuse detection, but still different, and generally

[2]https://www.perspectiveapi.com

more specific, e.g., insult or cyberbullying detection. The work of Balci & Salah differs from ours in the way they solve the problem, as they focus on the users' profiles and behaviors: these data are not available in our case, so we only use the published messages. Our previous work [17] is completely based on the textual content of the messages, whereas the one presented here ignores it, and relies only on a graph-based modeling of the conversation, which is completely new in this context. Another important methodological difference with the literature is that almost all content-based methods rely on manually constituted linguistic resources, which makes them difficult to transpose to another context (different languages or online community). By comparison, our present approach is completely language independent, as it does not use the textual content (apart from user names). The third difference is that almost all methods from the literature consider messages independently, when we use sequences of messages forming conversations. Finally, we use a classic classifier to determine if a message is abusive, which means that our approach requires much less training data than the deep learning methods that we mentioned earlier.

B. Network Extraction From Conversation Logs

Although a major part of the methods proposed to address the abuse detection problem focus on the content of the exchanged messages, it appears that a user with previous exposure to automatic moderation techniques can easily circumvent them [13]. To avoid this issue, a solution would be not to focus on the textual content, but rather on the interactions between the users through these messages. For instance, the number of respondents to a given message appears frequently as a classification feature in the literature, e.g., as in [18]. But graphs constitute a more natural paradigm to model such relational information, under the form of so-called conversational networks, which represent the flow of the conversation between users. Such networks have the advantage of including the mentioned feature (number of respondents), but also much more information regarding the way users interact. We adopted this approach in [3], which is the first attempt at using such graph-based conversation models to solve a general abuse detection problem. Our present work is an extension of this method, essentially on two aspects: we experiment with several variations of our graph-extraction process, and we consider much more graph-based features.

This section reviews methods proposed in the literature for the extraction of conversational networks. We do not narrow it to the abuse detection context, as [3] would be the only one concerned. Even so, there are not many works dealing with the extraction of conversational networks. This may be due to the fact that the task can be far from trivial, depending on the nature of the available raw data: it is much harder for chat logs than for structured messages board or Web forums, for instance. In a multiparticipant chat log it is frequent to see multiple disjointed conversations overlapping. There is no fixed topic although some chatrooms have a general purpose. There is also no built-in mechanism to specify the message someone is responding to. Finally, in most Internet relay chat (IRC) chat logs, there is no enforcement mechanism to ensure that users have only one nickname.

Mutton [20] proposes a strategy to extract such a network from IRC chat logs. The goal is to build a tool to visualize user interactions in an IRC chat room over time. The author uses a simple set of rules based on direct referencing (i.e., when a user addresses another one by using his nickname), as well as temporal proximity and temporal density of the messages. In our own work, we adapt and expend on some of these rules, whereas certain cannot be applied. Specifically, while in a regular IRC channel timestamps are indeed useful to determine intended recipients of a message, in our case they are basically irrelevant.

Osesina *et al.* [21] build on the work of Mutton using response-time analysis, which assumes that both temporal proximity and the cyclical nature of conversations can be used to perform edge prediction. The authors also use the content of the communications to build a word network, and then assign edges between users based on the keywords they use and the presence of these keywords in word clusters. Finally, by combining these two approaches with direct addressing, they achieve impressive performance in edge prediction with regard to a manually extracted network, both in terms of edge existence and edge strength. It is worth noticing the significant computational requirement for large chat logs. Besides the targeted task itself, the main difference with our approach is that this one is strongly content-based.

Gruzd and Haythornthwaite [22] push the usage of direct referencing further by developing methods of name discovery. The data they work on come from a bulletin board which shares some similarities with regular chat: linear stream of messages with possibly intertwined discussion threads. By comparing a network extracted through their name discovery method, to a chain network based on temporal proximity, they show that their approach is better suited to detect social network links. Useful takeaways of their method are: the use of neighboring words (for instance, Dr., Pr., and Jr, are often seen in proximity of person names, whereas Street and Ave are often near location names), capitalization, and the position of words within the document (e.g., their sample of posts often end with a user's signature because a bulletin board does not have the ephemeral nature of chatrooms). However, these differences between the two media also makes this method unsuitable to our data.

Çamtepe *et al.* [23] experiment on the detection of groups of users in chat logs, collected from three different chatrooms in the Undernet IRC network. They first build a matrix containing the numbers of messages posted by each user at each considered time step. It can be considered as a low-resolution view of the logs—it retains information about temporal proximity but looses sequential information. They then perform singular value decomposition on this matrix, in order to ease the identification of clusters of interacting users (i.e., conversations). They extract an approximation of the conversational network from this partition, by representing each cluster by a clique. They validate their approach by manually extracting the actual conversational network directly from the logs, and comparing their structures. The main difference with our situation is that

the conversational graph is only seen as a way to validate the user group detection method: we want to use it as a model of the interactions.

Forestier *et al*. [24] tackle the extraction of networks from online forums. While the structure of conversations is explicitly represented on certain platforms, this it not the case there: a thread is represented as a flat sequence of messages. This makes it challenging to determine the intended recipient of a message. The authors show that by using a combination of grammatical analysis and Levenshtein distance computation for substrings, they can often ascertain who talks to whom. The resulting network can then be used to analyse the role of users in the community. The main difference with our method is that we ignore the content of messages.

Travassoli *et al*. [25] explore different methods to extract representative networks from group psychotherapy chat logs. One of them includes fuzzy referencing to mitigate effects of misspelled nicknames, and rules for representing one-to-all messages. The bulk of the methods uses static patterns of exchanges to predict a receiver. Their system shows a good agreement score with a human annotator. It is worth noting, though, that these logs are substantially different from ours: the psychotherapy sessions have well defined boundaries and a limited number of participants. This prevent the transposition of this approach to our problem.

Sinha and Rajasingh [26] use only direct referencing, but with the same fuzzy matching strategy as in [25], in order to extract a network representing the activity in the `#ubuntu` IRC support channel. This method manages to expose high level components of the Ubuntu social network, which in turn allows for the qualification of user behaviors into specific classes such as beginner or expert. This method of building user models can be very interesting when the data describing the users are scarce, as is the case on IRC where everyone can join and there is no requirement to register. While it does not allow for the direct classification of individual messages, the behavior information can be useful as a supporting feature in a text classification task.

Anwar and Abulaish [27] build a framework allowing to query user groups and communities of interest, based on the data extracted from the computers of suspects during a criminal investigation. They use a social graph extraction method that relies both on the presence of communication between users and the overlap between the content of the messages they exchange, in order to assign weights to the edges of the network. They then experiment with various forms of community detection (i.e., graph partitioning) to identify groups of users in this network. However, this method assumes that the corpus contains a variety of topics allowing to discriminate the groups, which is not necessarily the case for us, since our logs are thematically dominated by the video game hosting the chatroom.

An interesting task where conversational networks can be used is the detection of controversial discussions. Garimella *et al*. [28] show that the predefined types of interactions allowed by Twitter can be used to build networks that highlight the presence of polarized groups of users. They extract all tweets matching a given hashtag around the time a specific event happens, then detect an endorsement link thanks to Twitter's retweet feature. The resulting graph is then partitioned and analyzed using a controversy measure. In our context it is difficult to adopt this approach, as endorsement information is not immediately available and would have to be inferred from message content.

The methods proposed in the literature mainly rely on the content of the exchanged messages. By comparison, our method only focuses on the presence/absence of communication between the users, i.e., on the dynamics of the conversation and its structure. Some methods also rely on specific functionalities of the studied platforms (e.g., answers explicitly addressed to a user), which are absent from our own data.

III. Methods

In this section, we describe the methods that we propose to compute the features later used in the classification task to separate abusive and nonabusive messages. We first present how we extract conversational networks from series of raw chat messages (Section III-A), before describing the topological measures that we use to characterize these networks (Section III-B).

A. Network Extraction

We extract networks representing conversations between users through a textual discussion channel. They take the form of weighted graphs, in which the vertices and edges represent the users and the communication between them, respectively. An edge weight corresponds to a score estimating the intensity of the communication between both connected users. We propose two variants of our method, allowing to extract undirected versus directed networks. In the latter case, the edge direction represents the information flow between the considered users. Note that each network is defined relatively to a targeted message, since the goal of this operation is to provide features used to classify the said message.

The method that we use to extract the networks representing the conversations in which each message occurs has three steps that we describe in detail in this section. First, we identify the subset of messages that we will use to extract the network (Section III-A1). Second, we select as nodes a subset of users which are likely receivers of each individual message (Section III-A2). Third, we add edges and revise their weights depending on the potential receivers (Section III-A3). We describe and discuss the resulting conversational graphs in Section III-A4

1) Context Period: Our first step is to determine which messages to use in order to extract the network. For this purpose, we define the context period, as a sequence of messages. Fig. 1 shows an example of context period, representing each message as a vertical rectangle. Note that time flows from left to right in Fig. 1. This sequence is centered on the targeted message (in red), and spans symmetrically before (left side) and after (right side) its occurrence. Put differently: we consider the same number of past and future messages. The networks extracted from the context period contain only

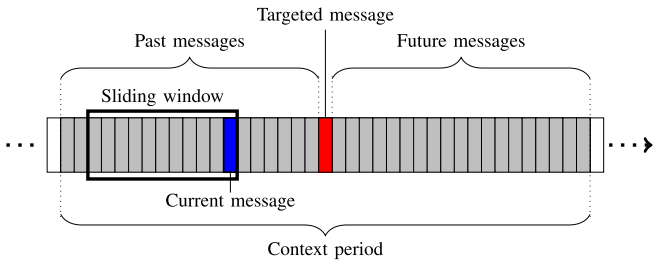


Fig. 1. Sequence of messages (represented by vertical rectangles) illustrating the various concepts used in our conversational network extraction process. Figure available at 10.6084/m9.figshare.7442273 under CC-BY license.

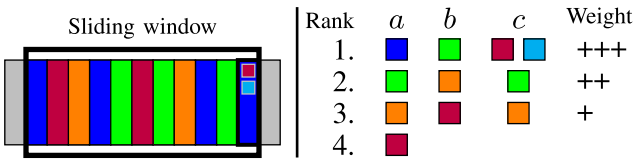


Fig. 2. Example of sliding window (left) and computation of the corresponding receivers' scores (right). Each color represents a specific user. Each message in the window is filled with the color of its author (left), whereas the small squares represent direct references to users. *a*, *b*, and *c* columns represent different steps of the computation (right) (see text). Figure available at 10.6084/m9.figshare.7442273 under CC-BY license.

the vertices representing the users which posted at least once on this channel, during this period.

Besides the network extracted over the whole context period (before and after the targeted message), which we call the Full network, we also consider two additional networks. We split the period in the middle, right on the targeted message, and extract one network over the messages published in the first half (Past messages), called Before network, and one over the other half (Future messages), called After network. Both of those smaller networks also contain the targeted message. For a prediction task, i.e., when using only past information to classify the targeted message, one would only be able to use the Before network. However, in a more general setting, all three networks (Before, After, and Full) can be used.

2) Sliding Window: In order to extract a network, we apply an iterative process, consisting in sliding a window over the whole context period, one message at a time, and updating the edges and their weights according to the process described next. The size of this sliding window is expressed in terms of number of messages, and it is fixed. It can be viewed as a focus on a part of the conversation taking place at some given time. It is shown as a thick black frame in Fig. 1. We call current message the last message of the window taken at a given time (represented in blue), and current author the author of the current message.

The use of such a fixed-length sliding window is a methodological choice justified by four properties of the user interface of the considered discussion channel: 1) at any given time, the user can see only up to 10 preceding messages without scrolling; 2) when a user joins a channel, the server sends him only the last 20 messages posted on the channel; 3) it is impossible for a user to scroll back the history further than 20 lines; and 4) the user interface masks join and part events by default, whereas in typical chat clients the arrival and departure of users are shown by default. Thus, at some given time, a user only has access to a limited knowledge regarding who is participating in the conversation. As explained later, we use this value of 20 messages as an upper bound, and experiment with different sliding window sizes.

3) Weight Assignment: Our assumption is that the current message is destined to the authors of the other messages present in the considered sliding window. Based on this hypothesis, we update the edges and weights in the following way. We start by listing the authors of the messages currently present in the sliding window, and ordering them by their last posted message. Only the edges toward the users in that list will receive weight. This choice is also due to the user interface constraints: *a priori*, a user cannot reliably know which users will receive a given message. Furthermore, the data we have do not allow us to directly determine channel occupancy at the time a message is posted.

Fig. 2 (left) displays an example of sliding window, in which the colors of the messages (vertical rectangles) represent their authors. So, in this specific case, four different users participate in the conversation. Ordered from latest to earliest, these are: blue (author of the current message, i.e., the rightmost in the window), green, orange, and red. This list of users is noted *a* in Fig. 2 (right). Obviously, a user is not writing to himself, so we remove the current author from the list, resulting in list *b*. The use of such an ordered list is justified by the assumption of temporal proximity, which appears commonly in the literature concerned with the extraction of conversational networks ([20], [21], [23]). It states that the most recent a message, the most likely its author to be the recipient of the current message.

The user interface allows us to explicitly mention users in a message by their name, and moreover the game prevents the users from changing their name: we need to take these properties into account. It is also a common assumption that the presence of direct referencing increases the likelihood that the referred person is the intended recipient of the message. To reflect this in our process, we move the users directly referenced in the current message at the top of the list. If some users are directly referenced although they have not posted any message in the considered window, they are simply inserted at the top of the list. In Fig. 2, direct references are represented as small colored squares located in the current message. There are two of them in our example, referring to the purple and cyan users. The former has one post in the window, so he is moved from the third to the first rank in the list. The latter did not post anything in the window, so he is inserted at the first position. This results in what we call the list of receivers, which appears as list *c* in Fig. 2.

We now want to connect the current author to the receivers constituting our ordered list. Our choice to create or update edges toward all users in the window even in case of direct referencing is based on several considerations. First, directly referencing a user does not imply that he is part of the conversation or that the message is directed toward him: for instance, his name could just be mentioned as an object of the sentence. Second, there can be multiple direct references in

a single message (as in our example). Third, in online public discourse, directly addressing someone does not mean he is the sole intended recipient of the message. For instance when discussing politics, a question directed toward someone can have as a secondary objective to have the target expose his stance on an issue to the other participants.

We also want to adjust the strength of each of these connections depending on the rank of the concerned receivers: the higher the rank, the stronger the interaction. For this purpose, each receiver is assigned a score, which is a decreasing function of both his rank i in the list and of the length N of this list (as reflected by the number of $+$ signs in Fig. 2). We propose three different scoring functions, defined so that the assigned weights sum to unity.

1) *Uniform:* Each receiver gets the same weight, defined as

$$f_U(i) = \frac{1}{N}. \tag{1}$$

2) *Linear:* The score decreases as a linear function of the rank

$$f_L(i) = \frac{N-i}{\sum_{j=1}^{N} j}. \tag{2}$$

3) *Recursive:* The first receiver gets 60% of the total weight, and the rest of them share the remaining 40% using the same recursive 60%–40% split scheme

$$f_R(i) = \begin{cases} 0.6 \times 0.4^{i-1}, & \text{if } 1 \leq i < N \\ 0.4^{i-1}, & \text{if } i = N. \end{cases} \tag{3}$$

As an illustration, Fig. 3 displays the scores assigned by these three strategies for $N = 10$, as functions of the receiver's rank. The Uniform strategy f_U (in red line) assumes that the content of the communication is not really important, and that the goal of the current author is just to have the message seen by as much people as possible. It, therefore, places very little importance on temporal proximity or direct referencing. The Recursive approach F_R (in blue) gives the most importance to direct referencing and temporal proximity, with scores dropping fast when the receiver is not directly referenced or the author of the immediately preceding message. Finally, the Linear approach f_L (in green) also places the most importance on temporal proximity and direct referencing, but in a less contrasted way, since it assigns higher scores (compared to f_R) to receivers located at the bottom of the list. We later compare these three strategies during our experiments, in order to determine whether it is worth exploring more advanced scoring functions, or if the difference in performance is not significant enough to justify this.

We can then update the graph by creating an edge between the current author and each user in the receiver list. We consider two possible approaches, leading to an undirected versus a directed network. In the latter case, the edge is directed from the current author toward the receiver, in order to model the communication flow. Each newly created edge is assigned a weight corresponding to the receiver's score. If this edge already exists, we increase its current weight by the said score. Fig. 4 shows the result of this update based on our previous example from Fig. 2, for the extraction of an undirected

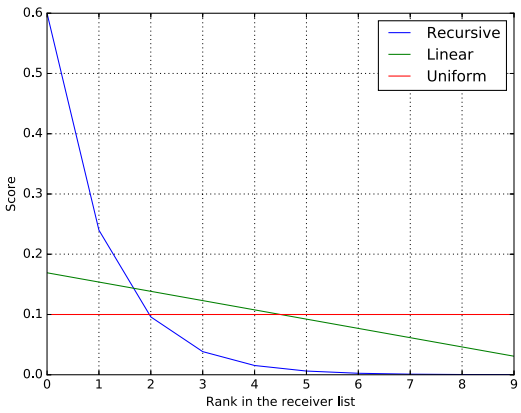


Fig. 3. Scores assigned by our three scoring functions f_U, f_L, and f_R for a receiver list containing 10 users.

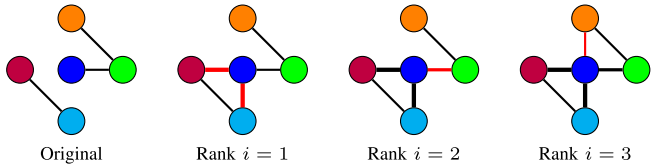


Fig. 4. Update of the edges and weights of the conversational graph corresponding to our ongoing example. The first graph displays the state before the update, and each remaining one corresponds to one rank in the receiver list. Figure available at 10.6084/m9.figshare.7442273 under CC-BY license.

network. The first graph represents the network before the update. It already contains some edges though, resulting from some previous processing. The remaining graphs of Fig. 3 represent the changes corresponding to the ranks appearing in the receiver list: first position (purple and cyan users), second position (green line), and third position (orange line). Red edges represent the edges being modified or created. If we were extracting a directed graph, then the new edges would be directed outward from the central blue vertex.

Once the iterative process has been applied for the whole context period, we get what we call the Full network. As mentioned before, for testing matters we also process two lesser networks based on the same context: the Before and After networks are extracted using only the messages preceding and following the targeted message, respectively, as well as the targeted message itself.

4) Extracted Networks: Fig. 5 shows a real-world example of the three conversational networks obtained by applying our extraction method to an abusive comment belonging to our dataset. They are obtained based on a context period of 200 messages, a sliding window of 10 messages, and are undirected. The isolates (disconnected vertices) present in the Before and After networks correspond to users present in the context period, but active only after or before the targeted message, respectively. The red vertex corresponds to the author of the targeted message, which we call the targeted user. One can see that the users involved in the conversation, as well as the location of the targeted user in this conversation, undergo some dramatic changes after the abuse.

Generally speaking, two vertices are connected in our networks if they are supposed to have a direct interaction. Thus, if only one conversation occurs during the considered context

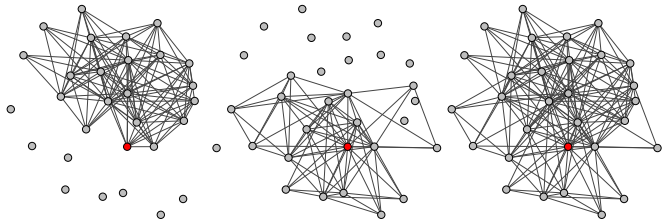


Fig. 5. Example of the three types of conversational networks extracted for a given context period: Before (left), After (center), and Full (right). The author of the targeted message is represented in red dot. For readability reasons, weights and directions have been omitted. Figure available at 10.6084/m9.figshare.7442273 under CC-BY license.

period, we expect the network to be rather cliquish. It seems possible to have several communities, i.e., several loosely connected dense subgraphs, if certain users completely ignore some other ones, for some reason. However, the smoothing induced by our use of a sliding window is likely to hide this type of behavior, especially if the window is large. The presence of a community structure could also occur if several distinct conversations take place during the considered context period. However, this can happen only if the number of common users between the conversations is small compared to the network size (otherwise, the communities will be indistinguishable). Due to the relatively dense nature of the networks (when ignoring isolates), we think weights are likely to be an important information, allowing to separate accidental edges from relevant ones. The edge direction allows distinguishing unilateral and bilateral interactions, so it could help identify certain types of conversations with atypical structure (e.g., one-way communication).

B. Features

The classification features that we consider in this paper are all based on topological measures allowing to characterize graphs in various ways. We process all the features for each of the 3 types of networks (Before, After, and Full) described in Section III-A.

We adopt an exploratory approach and consider a large range of topological measures, focusing on the most widespread in the literature. Some of these measures can optionally handle edge directions or edge weights: we consider all practically available variants, in order to assess how informative these aspects of the graph are relatively to our classification problem.

One can distinguish topological measures in terms of scale and scope. The scale depends on the nature of the characterized entity: vertex, subgraph, or graph. In our case, we focus only on vertex- and graph-focused measures: the former allows focusing on the author of the targeted message, whereas the latter describes the whole conversation, but we do not have any subgraph to characterize. The scope corresponds to the nature of the information used to characterize the entity: microscopic (interconnection between a vertex and its direct neighborhood), mesoscopic (structure of a subgraph and its direct neighborhood), and macroscopic (structure of the whole graph).

In the rest of this section, we describe these measures briefly: first the vertex-focused ones (Section III-B1), then the graph-focused ones (Section III-B2). For each measure, we give a generic, graph-theoretical definition, before explaining how it can be interpreted in the context of our conversational networks.

1) Vertex-Focused Topological Measures: These measures allows characterizing only a single vertex. We compute them all for the vertex corresponding to the author of the targeted message (represented in red in Fig. 5).

a) Microscopic measures: We start with the measures which describe a vertex depending on its direct neighborhood. In our context, this amounts to characterizing the position of some user depending on its direct interlocutors. In the case of a conversation involving a very small number of persons, it is likely all of them interact directly, and so these measures can also help describing the conversation itself.

The degree centrality is a normalized version of the standard degree [46], [50], which corresponds itself to the number of direct neighbors of the considered vertex. In a directed graph, one can distinguish an incoming and an outgoing degree centrality, focusing only on the incoming and outgoing edges of the vertex, respectively. In our case, it can be interpreted as the number of users that have exchanged (undirected version), received (outgoing), or sent (incoming) messages to the author, respectively. We use both undirected and directed variants of the degree centrality.

The generalization of the degree to weighted networks is called the strength [47]. The strength centrality is based on the sum of the weights of the edges attached to the considered vertex. Like the degree, it is possible to use incoming and outgoing versions if the network is directed. In our conversational graph, compared to the degree, the strength takes into account the frequency of the interactions. This allows accounting for certain situations ignored by the degree centrality. For instance, a user can have a few interlocutors, but still be central if he exchanges a lot with them. We use both undirected and directed variants of the strength centrality.

The local transitivity (or clustering coefficient) [48] corresponds to the proportion of edges between the considered vertex's neighbors, relatively to what this number could be if all of them were interconnected. It ranges from 0 (no interneighbor edge at all) to 1 (the vertex and its neighborhood form a clique). In our context, a high transitivity indicates that the user belongs to a single conversation, in which most protagonists exchange messages. On the contrary, a low transitivity denotes some form of segmentation: either the user participates in several distinct conversations, or some of his interlocutors ignore each other. We use the unweighted original version and the weighted variant presented in [47].

Burt's constraint [49] measures how redundant the neighbors of the vertex of interest are. It is based on the idea that a vertex located at the interface between several independent groups holds a position of power. Burt's constraints measure this level of independence through a nonlinear combination of the number of connections between the neighbors. A high value indicates how embedded the vertex is in its

neighborhood. In our case, this can help distinguishing users depending on the number of conversations they are involved in, if we suppose a conversation corresponds to a clique-like structure. We use both unweighted and weighted variants of Burt's constraint.

b) Macroscopic measures: The measures harnessing the entirety of the graph structure form the largest group. In our context, they allow characterizing the position of a vertex relatively to the whole context period (Full) or to one of its halves (Before and After).

So-called spectral measures are based on the spectrum of the graph adjacency matrix, or of a related matrix. The eigenvector centrality [34] can be considered as a generalization of the degree, in which instead of just counting the neighbors, one also takes into account their own centrality: a central neighbor increases the centrality of the vertex of interest more than a peripheral one. Central vertices tend to be embedded in dense subgraphs. We use the (un)weighted and (un)directed variants of the measure (so: four variants in total).

One limitation of the eigenvector centrality is that if the graph is directed and not strongly connected, certain vertices systematically get a zero centrality, whatever their position. Several modifications have been proposed to handle this situation. The hub and authority scores [35] are two complementary measures processed through the hyperlink-induced topic search algorithm. They solve the issue by splitting the centrality value into two parts: one for the incoming influence (authority), and the other for the outgoing one (hub). We use the (un)weighted directed variants of both hub and authority scores.

The alpha centrality (or Katz centrality) [36], [51] solves the same problem by assigning a minimal positive centrality value to all vertices. In addition, it allows attenuating the influence of distant vertices during the computation. We use the (un)weighted directed variants of this measure. The power centrality [37] generalizes both the eigenvector and alpha centralities. In particular, it allows a negative attenuation. The implementation we use only works for unweighted directed (UD) graphs.

The pagerank centrality [38] can be seen as a variant of the Katz centrality. One limitation of the later is that when a central vertex has many outgoing edges, all of them receive all its influence, as if they were its only recipient. The pagerank centrality includes a normalization allowing to model the dilution of this influence. We use the (un)weighted and (un)directed variants of this measure.

Compared to the other spectral measures, the subgraph centrality [39] defines the notion of reachability based on closed walks rather than simple walks. Put differently, the other spectral measures consider that the vertex of interest influences (resp. is influenced by) some other vertex if a walk exists to go to (resp. come from) this vertex. The subgraph centrality requires both, and it uses an attenuation coefficient to give less importance to longer walks. The implementation we use only deals with undirected unweighted graphs.

In our conversational graph, we expect that a user participating a lot in the conversation will be central, and even more so if there are several conversations and he is participating in the main one. It is difficult to predict which ones of these slightly different spectral measures will be the most appropriate to our case, which is why we included all of the available ones.

Another group of macroscopic measures is based on the notions of shortest path or geodesic distance (i.e., the length of the shortest path).

The betweenness centrality [40] is related to the number of shortest paths going through the considered vertex. In communication networks such as ours, it can be interpreted as the level of control that the user of interest has over information transmission. We use the (un)weighted and/or (un)directed variants of this measure.

The closeness centrality [41] is related to the reciprocal of the total geodesic distance between the vertex of interest and the other vertices. It is generally considered that it measures the efficiency of the vertex to spread a message over the graph, and its independence from the other vertices in terms of communication. The eccentricity [42] is related to the closeness centrality, but it is not a centrality measure. On the contrary, it quantifies how peripheral the vertex of interest is, by considering the distance to its farthest vertex. By comparison to the closeness centrality, there is no reciprocal involved, and it uses the maximum operator instead of the sum. In our case, both measures indicate how involved the considered user is in the conversation(s), as they directly depend on how directly connected he is to the other users. In particular, we expect important changes in the Before and After graphs to reflect a significant modification of the user's role in the conversation. For the closeness centrality, we use the (un)weighted and (un)directed variants, but for the eccentricity, we only have access to the unweighted (un)directed variants.

The last group of macroscopic measures is based on the notion of connectivity, i.e., whether or not a path exists between certain parts of the graph.

An articulation point (or cut vertex) is a vertex whose removal makes the graph disconnected, i.e., split it into several separate components [42]. We define a binary nodal feature indicating if the vertex of interest is an articulation point (1) or not (0). It could help describing whether the targeted user is bridging two separate groups of users in the conversation, possibly indicating that he caused a topic shift or that some of the users have left the conversation.

c) Mesoscopic measures: Mesoscopic measures rely on an intermediate structure to characterize a vertex. In our case, such a subgraph corresponds to a tightly knit group of users, and is likely to represent a conversation. So, this type of measure would allow characterizing the position of a vertex relatively to the various conversations taking place in the considered context period (provided there are several of them).

The coreness score [43] is based on the notion of k-core, which is a maximal induced subgraph whose all vertices have a degree of at least k. The coreness score of a vertex is the k value of the k-core of maximal degree to which it belongs. In our context, the coreness score is related to the number of participants of the largest conversation involving the user of interest. We use an undirected version of the coreness score, as well as two variants focusing on incoming and outgoing edges in directed networks.

We also take advantage of the within-module degree and participation coefficient, a pair of complementary measures defined relatively to the community structure of the graph [44]. We detect the community structure through the InfoMap method [52]. These measures aim at characterizing the position of a vertex at this intermediate level. The within-module degree (or internal intensity) assesses the internal connectivity. It evaluates how the degree of a vertex within his community relates to those of the other vertices from the same community. For us, it is an indicator of how involved the user is in his current conversation. The participation coefficient is concerned with the external connectivity: it is based on the number and quality of the connections that the vertex has outside of his own community. In our case, a high value could indicate either someone holding a mediation position, in the case of a single conversation involving several groups of users, or someone participating in several conversations. We use the original undirected variants of these measures, as well as the directed variants proposed in [53] to focus on incoming and outgoing edges.

One limitation of the participation coefficient is that it mixes several aspects of the external connectivity: the number of external connections, the number of concerned external communities, and the distribution of these connections over these communities. To solve this issue, three measures were proposed in [45] to separately assess these three properties. They are, respectively, called external intensity, diversity, and heterogeneity. The available variants are all unweighted, but allow handling undirected, incoming, and outgoing edges.

2) Graph-Focused Topological Measures: A simple way to obtain graph-focused measures is to consider a vertex-focused measure and compute some statistic over the vertex set of the graph. This is what we do for all of the 21 measures described in Section III-B, by averaging them over the whole graph. This also holds for all the variants (weighted and/or directed) of these measures. But there are also measures defined specifically for the graph scale: like before, we distinguish them based on their scope.

a) Microscopic measures: First, we use very classic statistics describing the graph size: the vertex and edge counts. We also compute the density, which corresponds to the ratio of the number of existing edges to the number of edges in a complete graph containing the same number of vertices. In other words, the density corresponds to the proportion of existing edges, compared to the maximal possible number for the considered graph. In our context, these measures allow assessing the number of users considered in a context period (vertex count), and the general intensity of their communication during this period (edge count). The Density can be viewed as a normalized edge count that is more likely to be useful when comparing graphs of different sizes.

The global transitivity (or global clustering coefficient [31] is the graph-focused counterpart of the Local Transitivity. It corresponds to the proportion of closed triads among connected ones, where a closed triad is a three-clique (i.e., a triangle) and a connected triad is a subgraph of three vertices containing at least two edges. This proportion measures the prevalence of triadic closure in the graph. In our context, it assesses how likely two users communicating with the same person are to directly exchange messages themselves. We only have access to the undirected unweighted version of this measure.

The reciprocity [32] is defined only for directed graphs. It corresponds to the proportion of bilateral edges over all pairs of vertices. In our networks, a low reciprocity would indicate that certain users do not respond to others.

The degree assortativity (or assortativity for short) [33] measures the homophily of the graph relatively to the vertex degree. The homophily is the tendency for vertices to be connected to other similar vertices (in this case: of similar degree). It is based on the correlation between the series constituted of all pairs of connected vertices. We use both directed and undirected variants of this measure. In our conversational networks, this measure could help detect situations where users do not participate to the conversation at the same level.

b) Macroscopic measures: A number of macroscopic measures are connectivity-based. The weak component count corresponds to the number of maximally connected subgraphs. In such a subgraph, there is a path to connect any pair of vertices. For our conversational networks, this could correspond to a conversation, whose participant do not necessarily talk directly to each other. However, due to the use of a sliding window, we expect our graphs to be connected (i.e., only one weak component), even if by very weak edges. In this case, a conversation is more likely to correspond to other substructures based on more relaxed definitions, such as cliques or communities. For directed graphs, we also consider the strong component count: a strong component is similar to a weak one, except it is based on directed paths. We suppose that, in our networks, we are more likely to get several strong components, since users do not necessarily exchange in a bilateral way.

The cohesion (or vertex connectivity) of a graph corresponds to the minimal number of vertices one needs to remove in order to make the graph disconnected (i.e., have several components) [29]. The adhesion (or edge connectivity) is similar, but for edges. In our conversational networks, these measures can be related to the level of participation to the considered conversation: the higher their values, and the higher this level. But high values can also denote the presence of several distinct conversations in the context period. Both measures are defined for directed networks.

As mentioned before when describing the nodal measures, we check whether the targeted user is an articulation point. We also compute the articulation point count, i.e., the total number of articulation points in the graph. This measure is related to the Cohesion, since there are no articulation point if the Cohesion is larger than 1. The implementation we use handles only undirected graphs. In our context, the number of articulation points could be related to the presence of several conversations (articulation points corresponding to gateway users between them). It could also reflect situations where a conversation lasts a very long time, and some groups of users loose interest and get disconnected from the active users. Another possibility is the occurrence of a flood-type situation: a user sends a flurry of messages into the channel to kill a

conversation, then leave, and a different group of users later takes possession of the channel to start its own conversation.

We also use three distance-related measures. The first is the diameter, which corresponds to the largest distance found in the graph, i.e., the length of the longest shortest path. It also corresponds to the largest eccentricity over all vertices. We use (un)weighted and (un)directed variants. The second is the radius, which is the smallest eccentricity over all vertices. We use its undirected, incoming and outgoing variants. The third is the average distance, which is the average length of the shortest paths processed over all pairs of vertices. We use its unweighted (un)directed variants. In our networks, the distance is related to the separation between users, in terms of interaction. A large Diameter means that a user can be many intermediaries away from exchanging directly with another user. This could be caused, for instance, by the occurrence of several distinct conversations in the considered context period, or by a very long conversation loosing and gaining users through time. This observation also holds for the radius and average distance, which provide a slightly different perspectives on the same aspect of the graph.

c) Mesoscopic measures: We process the total clique count in the network, where a clique is a complete induced subgraph. As mentioned eralier, this can be related to the number of conversations occurring in the context period, or to number of subgroups of users participating in the same conversation.

Like before with vertex-focused measures, we use the InfoMap algorithm to detect the community structure [52]. Based on this partition, we compute two measures: the community count and the modularity [30]. The latter assesses the quality of the detected community structure, i.e., how internally cohesive and externally disconnected the communities are. We use both weighted and unweighted variants of the Modularity.

IV. Experiments

This section describes our experimental setup and results regarding the automatic detection of abusive messages in chat logs. In Section IV-A, we present our dataset and the general architecture of our classification system. Because we expect some of our features to be redundant, we conduct a correlation study of our feature set in Section IV-B. We present general results and the effect of our various graph extraction parameters in Section IV-C. In Section IV-D, we investigate the temporal aspects of the system –specifically what happens when we train our models based on the features extracted from only one of the three graphs (Before, After, or Full), or some of their combinations. We then examine the importance of weight and directionality in Section IV-E, before investigating the potential for computational optimization through feature selection in Section IV-F. Finally, in Section IV-G, we compare the performance obtained using the best configuration of our framework with the selected baselines.

A. Experimental Setup

We have access to a database of 4 029 343 messages that were exchanged by the users of the browser-based multiplayer game SpaceOrigin, a French-language massively multiplayer online game. In this database, 779 messages have been flagged by one or more users as being abusive, and subsequently confirmed as abusive by the human game moderators: they constitute our abuse class. Each message belongs to a unique communication channel. A total of 226 distinct users have authored these abusive messages. We further extract 2000 messages at random from the messages not confirmed as abusive, to constitute the nonabuse class. Note that all the results we discuss in this paper are relative to the abuse class.

We previously experimented with this dataset in [3] and [17]. However, since then we have detected certain inconsistencies in the database, preventing us from retrieving the context of certain messages. We cannot apply our classification method to them, so we discard them for the work presented here. Note that this concerns both classes. Moreover, our tests show that removing those samples does not significantly impact our previous performances. The resulting dataset is constituted of 1890 messages in the nonabuse class and 655 messages in the abuse class. Fig. 6a shows the distribution of abuse cases by user. It suggests that most abusive users need only a few warnings before mending their ways, but it also shows that some users are exceptional in the number of abuses they commit.

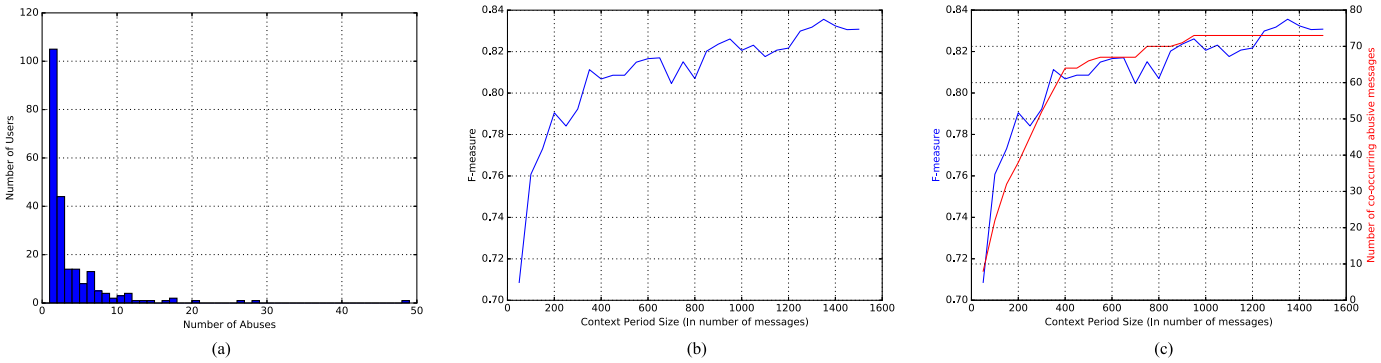
Because of the relatively small dataset, we set up our experiment for a tenfold cross-validation. We split the dataset into 10 same-sized parts containing the same ratio of abusive to nonabusive messages. We use a 70%-train / 30%-test split, which means, for each run of the cross-validation, the train set is composed of 7 of those parts while the test set is composed of the remaining 3. We use Python-iGraph [54] to extract the conversational networks and process the graph-based features for each message. As a classifier, we use a support vector machine (SVM), implemented in the Sklearn toolkit [55] under the name C-support vector classification.

We mainly experiment with 4 different sets of features: Full, Before, After, and All. For a given message, Before, After, and Full correspond to all the topological measures computed for the Before, After, and Full graphs, respectively. All is the union of all three sets, i.e., it includes all topological measures for all three graphs.

In the remainder of this section, we occasionally provide computational time requirements. For context, the times that we provide correspond to single-threaded calculations performed on an Intel Xeon CPU E5-2620 v3s, clock speed 2.5-GHz and 15-MB cache.

B. Feature Dependence Study

Each considered topological measure was originally defined to characterize a graph in a specific, distinct way. So, in theory, they could all be independent for a given graph, and thus all necessary to describe it completely. But in practice, according to the structure of the considered graph, some of them can be statistically dependent, and, therefore, redundant. In order to get a better understanding of the way these topological measures behave on our conversational graphs, we compare all the computed features using Pearson's correlation coefficient.



(a) (b) (c)

Fig. 6. a) Distribution of the number of abuse cases by user. b) Classification performance (F-measure) as a function of the context period size, for a sliding window of 10 messages and using the *Recursive* weight assignment strategy (f_R). c) Classification performance (F-measure, in blue) and number of abuse occurrences in the context period (red), as functions of the context period size.

In our context, where these features are later fetched to a classifier, only the strength of the association is relevant, i.e., the absolute value of the correlation (not its sign). We identify clusters of highly correlated features using the `hclust` function of the `R` language, which implements a standard hierarchical cluster analysis method, with average linkage. We use the Silhouette measure [56] as a criterion to select the best cut in the produced dendrograms. To keep the description short, we only focus on the most interesting results.

A very small number of features are constant over all instances of the corpus, which means they have no discriminative power at all. For all three types of networks, the number of weak components is always 1, which means they are always (weakly) connected. This can be explained by our use of a sliding window: even if the context period contains two separate conversations, they will be connected, possibly by an edge of quasi-zero weight. In the After and Full graphs, the targeted user is never an articulation point. Moreover, in the Full graph, the number of articulation points is always zero. We already know that the graphs are connected, so this zero value means no single vertex removal can disconnect them.

A few features are quasi-independent, in the sense they display almost no correlation with any other feature. This is the case of certain variants of the power, subgraph, and alpha centralities. From this point of view, they differ from the other spectral measures, which are overall strongly correlated. Certain variants of measures focusing on connectivity (strong component count, adhesion, cohesion, and radius) are also independent, and it is the case for a number of mesoscale features too, all of them based on the community structure. The fact that these features are only weakly (if at all) correlated to the others makes them singular, in the sense they are the only ones to capture certain structural changes in the conversational graphs. But it does not imply they have any particular discriminative power regarding the classification task at hand. However, they must be closely monitored in the rest of our experiments, because they constitute good candidates.

The rest of the features forms highly correlated clusters, some of which are homogeneous in terms of measure variants, while some are heterogeneous. As explained in Section III-B, for each topological measure we consider several variants to define our features: directed vs. undirected, weighted vs. unweighted, averaged vs. individual. In our case, certain variants of the same measure are strongly correlated, which makes them redundant for our purpose. In particular, a very large number of measures, mainly distance- and community-based, have strongly correlated direction-based variants. This indicates that most of the time, considering the direction of the interactions between users does not bring any additional information. This effect is clearly much less marked for average-based and weight-based variants. Thus, unlike direction, weight seems like an important aspect of our graphs, and averaging measures over all vertices also seems to bring some relevant information.

Overall, we observe different behaviors, which cannot be explained only by the various characteristics of the features (micro/meso/macroscopic, un/directed, un/weighted, Before/After/Full graphs). This supports our decision to adopt an exploratory approach to identify the most appropriate features for our classification problem. The detected clusters of correlated features will be useful later to ease the interpretation of the classification results, as features belonging to the same cluster can be considered as interchangeable.

C. Impact of Graph Extraction Parameters

As explained in Section III-A, our graph extraction method has three important parameters: 1) size of the context period; 2) size of the sliding window; and 3) weight assignment strategy. In this section, we explore how the classification performance varies depending on these parameters. Our goal here is both to get a better understanding of the parameters role, and to identify the most appropriate values without having to use brute force.

As a reminder, the context period is the sequence of messages considered to classify the targeted message, and symmetrically built around this message. We expect it to have a strong effect on the classification performance, depending on its size. If it is too small, one can suppose it only includes a part of the conversation containing the targeted message, and, therefore, lacks some information necessary to make a proper decision regarding the abusive nature of this message. On the contrary, if it is too large, we assume it contains several conversations having nothing to do with the targeted message, which should also result in lower classification performance. In summary,

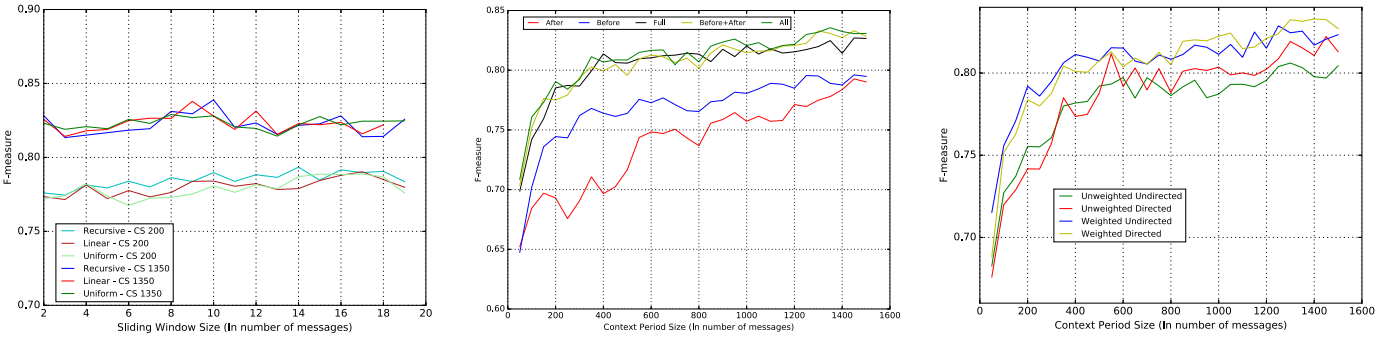


Fig. 7. a) Classification performance (F-measure) for the 3 considered weight assignment strategies (*Uniform, Linear* and *Recursive*) and 2 context period sizes (200 and 1,350 messages), as a function of the sliding window size. b) Classification performance (F-measure) as a function of the context period size (in messages), for the 4 considered feature sets (*Before, After, Full, All*) as well as a combination of the first two of them (*Before* + *After*). c) Performance (F-measure) obtained for the (un)directed and (un)weighted feature sets, as a function of the context period size.

when using a growing context period, we expect the classification performance to increase, then reach a plateau corresponding more or less to the typical duration of a conversation, and then decrease as the context period contains more and more noise (i.e., information not related to the targeted message).

Fig. 6b shows the evolution of the classification performance, expressed in terms of F-measure for the abuse class, as a function of the context period size, as for it expressed in numbers of messages. We fix the sliding window to 10 messages, and the recursive strategy f_R to assign weights. All available features (All feature set) are used during the classification. We choose these extraction parameters because earlier testing showed that the recursive strategy yielded the best performance, and this sliding window size provides a good tradeoff between a graph that would be very sparse and, therefore, not informative enough, and one that would be very dense and thus too noisy.

It appears that our assumption is only partially verified: the performance first increases with the context period size. However, it does not reach a plateau as expected, and, on the contrary, seems to go on increasing, albeit more and more slowly, as if it was logarithmically depending on the context period size. The maximal performance is obtained for a size of 1350 messages, but it is possible that even higher values can be obtained for larger context periods (which we did not check due to computational limitations). This means that our assumption regarding that large context periods would bring mainly additional noise is incorrect, because, on the contrary, they convey more relevant information concerning the classification task.

We manually investigate a sample of our dataset to understand this trend. From reading a number of conversation logs, it first appears that conversation boundaries are not well defined, and so there is no typical duration for a conversation: this can explain the absence of a plateau in the plot. Furthermore, we based our assumptions with regard to performance on the idea that an abusive message has a specific impact on what happens after it is posted. Specifically, the conversation would show markers of normality before the message occurs and quickly devolves after that. As it turns out, in conversations where an abusive message is found, the author of the abusive message usually has been around for a while, and the message that is actually flagged and confirmed as abusive is not his first suspicious message. We assume that the classifier can take advantage of this type of situations, therefore, invalidating our previous assumption that a large context period would only bring noise.

Note that more than one message of the abuse class can co-occur, i.e., can appear in the same context period, if it is large enough, which also supports our point. This is generally due to a single user sending multiple abusive messages in quick succession, or because the conversation devolves into name-calling following an initial abusive message. Fig. 6c shows the number of co-occurring abusive messages, as well as the F-measure performance, as functions of the size of the context period. There is a very strong match between both series, even if not perfect. This seems to back our assumption of the classifier taking advantage of the potential abuse cases happening around the targeted message. All these observations regarding the co-occurrence of abuse expose a couple of interesting perspectives: 1) user models can presumably yield features useful for classification and 2) a text-based model of the whole conversations would also likely be useful.

We now explore the impact of the window size and weight assignment strategy on the overall performance. Fig. 7a shows the evolution of our performances for two fixed sizes of context periods (200 and our previously obtained optimum of 1350). The maximal window size considered is 19, which corresponds to almost twice the default GUI limitation (cf. Section III-A2).

For our optimal context size, we obtain the best results for a window size of 9 and the linear assignment strategy, and for the window size 10 and the Recursive assignment strategy. Both of those strategies give greater importance to temporal proximity. Overall, there is no much difference between our weight assignment strategies. It seems that the specific values of the weights are not as important as their relative ordering. It is also worth noting that those window sizes are very close to the natural limitation of the GUI, which means they likely best capture the intended recipients of any given message.

D. Temporal Aspects

The results shown until now are all obtained using the All feature set, i.e., the features resulting from the calculation of

all topological measure variants for all three individual graphs (Full, Before, and After). However, using the Full or After graphs restricts the possible use cases for the system to tasks that do not require taking a decision as soon as the message to classify becomes available since the "future" context of the targeted message is taken into account. In order to have a system that is capable of doing so, we must investigate the impact of the Before features. Studying the features from the three individual graphs also allows us to get a better understanding of the system by providing a qualitative and accurate analysis of each part of the context.

Fig. 7b shows the results obtained for classifiers built using combinations of the available feature sets: After, Before, and Full correspond to each graph considered separately, whereas Before + After denotes the union of the Before and After sets, and All represents the set of all computed features (Before, After, and Full). The conversational graphs used for these experiments are extracted using the recursive assignment strategy f_R and a sliding window of 10 messages. Unless stated otherwise, we use these parameters in the rest of this paper as they match the best performance obtained during our greedy search of the parameter space (Section IV-C).

Since the main idea behind our approach is to detect the nature of a message based on the reaction it triggers in the community, it is not surprising to see that the After feature set (in red curve) reaches an acceptable performance level on this task. However, what is surprising is that by using only the Before feature set (in blue curve), the system performs much better on the same task (at least for small context periods). This suggests that the interactions occurring before the targeted message reveal more about its abusive nature than those happening after.

Nevertheless, when using large context periods, the performances obtained for Before and After get very similar. This indicates that what is important is not whether the messages used to extract the conversational graph precede or follow the targeted message, but rather how many of these messages are used. This supports our previous finding, regarding the fact that the context period size is the most important parameter of our graph extraction procedure. Moreover, based on the same observation, one could also think that the Before and After graphs convey approximately the same information, when considering large context periods, because the corresponding performances are roughly the same. However, this is disproved by the results obtained for the union of the Before and After feature sets (in yellow): the classification performance is noticeably higher, which means both types of graphs do not completely overlap, informationally speaking.

It is worth noticing that we get almost the same performance with the Full feature set as with Before + After. One could assume that the use of two distinct graphs built on either sides of the targeted message would help better characterizing it, compared to the Full feature set, which covers the same time span based on a single graph. Indeed, when extracting the latter, the sliding window passes through the targeted message, and is likely to smooth the potentially relevant topological changes occurring right around it. However, even if the performance gap between Before and After seems to widen when the context period gets larger, the difference is not clear, so this assumption is not verified. This means that the single Full graph is as approximately as informative as the joint use of both Before and After graphs. The latter option procures more flexibility in the possible application scenarios, but it contains twice as many features, and, therefore, requires roughly twice the computational time. This observation is confirmed when we consider the All feature set (in green), which contains all features for all three graphs. As expected, it performs best overall, since it is the union of all the other considered feature sets. However, the results are only marginally better than for Full and Before + After. This means that the information conveyed by the Full and Before + After feature sets essentially overlaps: using their union does not bring any noticeable performance increase.

E. Impact of Weights and Directions

We now investigate how considering the edge weights and directions in our features affects the classification performance. Based on the All feature set, we define four new feature sets, characterized by their focus on unweighted undirected (UU), UD, weighted undirected, and weighted directed measures, respectively. Concretely, each set includes the same group of core features, which are conceptually not concerned by the notion of weight or direction. This core is completed by features designed to consider or ignore weights or directions. For instance, the Clique Count is a core feature, whereas each one of the four variants of the diameter appears in a specific set.

Fig. 7c displays how the corresponding classification performance (in terms of F-measure) evolves as a function of the context period size. It appears that both weighted feature sets (blue and yellow) dominate their unweighted counterparts (green and red curves) over the considered interval. This seems to confirm our assumption from Section III-A, regarding the fact that weights can help discriminate between certain structures of conversations, and/or distinguish consecutive conversations. There is a similar effect for directions, but it is much weaker, as each directed feature set (red and yellow) only partially dominates its undirected counterpart (green and blue). This is consistent with our observation from Section IV-B, regarding the high correlation noticed for certain measures, between their undirected and directed variants. This indicates that the direction of edges is not as relevant as their weight relatively to the classification task at hands. Yet, the best performance is reached when using both weights and directions. If the additional computational cost is not too high (and it is generally not the case), it is, therefore, worth using directed features.

F. Feature Contributions

In order to estimate the discriminative power of our features with regard to this classification task, we use a recursive feature elimination method. It takes a given feature set as input, and outputs its subset of so-called top features (TFs). These are the minimal subset of features allowing to reach 97% of the performance obtained when considering the input

TABLE I

COMPARISON OF THE PERFORMANCES OBTAINED WITH THE FEATURE SETS (ALL, BEFORE, FULL, AND AFTER) AND THEIR SUBSETS OF TOP FEATURES (TF). THE TOTAL RUNTIME IS EXPRESSED AS day:h:min:s

Feature set	Number of features	Total Runtime	Average Runtime	F-measure
All	459	4:15:29:24	157.71 s	**83.89**
All-TF	10	53.92	0.02 s	**82.65**
Before	153	1:05:51:06	42.23 s	79.03
Before-TF	8	29.68	0.01 s	79.02
After	153	1:06:04:15	42.54 s	78.28
After-TF	11	1:30.72	0.04 s	76.01
Full	153	2:03:34:02	72.94 s	82.17
Full-TF	6	2:38.37	0.06 s	82.65

feature set. In order to identify these TFs, we apply an iterative method based on Sklearn. This toolkit allows us to fit a linear kernel SVM with the values of the input feature set, and provides a ranking of the individual features in that set, reflecting their relevance to the classification task. We then drop the least important feature, and train a new model using all the remaining features. We repeat this process until the classification performance reaches the targeted minimal threshold of 97% of the original F-measure score.

We first apply this recursive feature elimination process to the All feature set in order to identify the overall best features, then do the same with the Before, After, and Full feature sets, for comparison purposes. Table I presents the performances and computational time costs measured for each of these complete feature sets, as well as for their respective TF subsets. It appears that, for all four feature sets, using the TFs during the classification allows reducing the runtime by up to 4 orders of magnitude, while retaining at least 97% of the F-measure value, which is very interesting from an application perspective. It means that the longer features to process do not bring more discriminative power than the shorter ones, regarding the classification task at hand (at least for our dataset). The fourth column describes the average runtime by message, and shows that the classifier could operate in real-time when limited to the TFs.

It is important to notice that feature computation is by far the most computationally expensive step of our framework. By comparison, extracting the conversational graphs for the full corpus takes around 3 min, and performing the whole cross-validation (i.e., tenfold training and testing) only 1 min. The time required to compute our features depends on the size of the conversational graph, in terms of number of vertices and/or edges. The graph size is affected, in turn, by the number of users involved in the conversation (number of vertices) and the density of exchanged messages (number and weights of the edges). These characteristic are bounded by social and ergonomic (e.g., user interface) constraints, and can therefore be assumed as independent from the corpus size. The scalability of our method thus depends on that of the tool selected to perform the classification step: it is an SVM in this paper, but the end-user is free to use any other classifier instead.

As explained in Section IV-B, certain of our features are strongly correlated, which led us to identify clusters of interchangeable features. Studying the TFs would result in missing this information: we must consider their clusters instead. Table II displays the nine clusters corresponding to the TFs obtained for the All feature set. For matters of space, we discuss in detail this sole feature set only. Note that these clusters generally contain several variants of one (or more) topological measure(s), as indicated by the four last columns. For instance, Cluster 10 contains all variants of average weighted and UU eigenvector centrality for all three types of graphs (Before, After, and Full). Also note that a letter *G* in the scale column can either refer to a naturally graph-scale feature, or to a vertex-scale feature averaged over V (cf. Section III-B). For completeness, the proper TFs are represented in bold.

TABLE II

CLUSTERS CONTAINING THE TFS (IN BOLD) OBTAINED FOR THE ALL FEATURE SET. THE LETTERS IN THE GRAPH COLUMN STAND FOR BEFORE (B), AFTER (A), AND FULL (F). THOSE IN THE SCALE COLUMN MEAN GRAPH-SCALE (G) OR VERTEX-SCALE (N). THOSE IN THE WGHT. (WEIGHTS) AND DIR. (DIRECTIONS) COLUMNS STAND FOR: UNWEIGHTED OR UNDIRECTED (U), WEIGHTED (W), DIRECTED (D), INCOMING (I) AND OUTGOING (O).

Clust.	Measure	Graph	Wght.	Dir.	Scale
9	Clique Count	A/F	–	–	G
	Burt's Constraint	A/F	W	–	G
	Coreness Score	A/**F**	–	U/**I**/O	**G**
	Degree Centrality	A/F	U	U/I/O	G
	Strength Centrality	A/F	W	U/I/O	G
10	Assortativity	A/B/F	–	U/D	G
	Density	A	–	–	G
	Diameter	A	U	U/D	G
	Average Distance	A	U	U/D	G
	Radius	A	U	U	G
	Hub/Authority Scores	A/F	W	D	G
	Burt's Constraint	A/F	U	–	G/N
	Closeness Centrality	A	U	U/I/O	G
	Eccentricity	A	U	U/I/O	G
	Eigenvector Centrality	A/B/F	U/W	U	G
	PageRank Centrality	**A**/F	**U**/W	U/**D**	G/**N**
41	Degree Centrality	A/F	U	U/I/O	N
	Strength Centrality	A/**F**	**W**	U/I/**O**	**N**
49	**Vertex Count**	A/**F**	–	–	G
	Betweenness Centrality	A/F	U/W	U/D	G
110	Density	B	–	–	G
	Closeness Centrality	**B**	**W**	U/I/**O**	**G/N**
118	Average Distance	B	U	U/D	G
	Hub/**Authority Scores**	**B**	**W**	**D**	**G**
	PageRank Centrality	B	U/W	U/D	G/N
119	**Hub**/Authority Scores	A/**B**	**U**	**D**	**N**
172	**Reciprocity**	**A**	–	**D**	G
177	**Closeness Centrality**	**A**	**W**	**U**/O	G/**N**

Cluster 9 contains only micro- (degree, strength, Burt's constraint) and meso-scopic (Clique Count, Coreness) features describing the After and Full graphs. Moreover, all of them are graph-scale (as the vertex-scale measures are averaged over the graph). In contrast, Cluster 41 focuses on the same graphs and also contains the degree and strength, but as vertex-scale features this time. Put differently, Cluster 41 can be viewed as a vertex-scale counterpart of Cluster 9. This indicates that the microscopic characteristics of both the targeted vertex and the whole graph are relevant to our classification task.

Cluster 10 is very large and contains almost only graph-scale features. It focuses mainly on distance-based (diameter,

average distance, radius, closeness, eccentricity) and spectral (hub/authority, eigenvector, pagerank) macroscopic measures. Like the previous clusters, it essentially contains features computed on the After graph, but unlike them, it includes only a few features from the Full graph. Nevertheless, it appears as quite complementary of Cluster 9, in the sense it can be considered as its macroscopic counterpart.

Cluster 49 suggests that the betweenness of the After and Full graphs mechanically increases with their number of vertices. But more importantly, it identifies the Vertex Count, i.e., the size of the conversation after the targeted message, as one of its most discriminative aspects, relatively to our classification task. The interpretation of Clusters 172 and 177 is even clearer, as each focus on a single measure (reciprocity and closeness), uniquely for the After graph. The bilateral nature of the exchanges after the targeted message, as well as how direct these are, can, therefore, also be considered as very important for the classification.

Clusters 110 and 118 deal only with the Before graph. Cluster 110 includes variants of the weighted Closeness (both for the targeted vertex and in average) and Density. It can be considered as the Before counterpart of Cluster 177, which also focuses on the weighted Closeness but for the After graph. Cluster 118 contains distance-based and spectral macroscopic measures, mainly describing the whole graph. Thus, although much smaller, it can be seen as the Before counterpart of Cluster 10, semantically speaking.

Finally, Cluster 119 contains the unweighted hub and authority scores of the targeted vertex, for both After and Before graphs. It can be opposed to both Clusters 10 and 118, which also contain hub and authority for the After and Before graphs, respectively, but in their weighted and averaged versions.

Let us summarize our observations. A number of composite clusters describe the After/Full (Clusters 9, 10, and 41) and Before (Cluster 118) graphs at various scales and scopes. Two clusters focus more precisely on the Closeness, for the Before (Cluster 110) and After (Cluster 177) graphs. We assume that the classifier is able to take advantage of this to compare various aspects of the graphs, be it in terms of scale (Clusters 9 versus 41), scope (Clusters 9 versus 10) or time (Cluster 10 versus 118 and 177 versus 110). This means that 1) temporal aspects are useful for this classification task and 2) an abuse case is reflected by its impact on both the position of the abusive user in the graph and the overall aspect of the conversation.

Each remaining cluster (49, 119, and 172) focuses on a measure of the After graph, highlighting their contribution to class discrimination. We examine more thoroughly these features, by considering separately their distributions in the abuse and nonabuse classes. For the number of users in the conversation, it turns out these distributions are quite different: the vertex count is relatively homogeneous and centered around 40 for the abuse class, whereas its distribution is heterogeneous (closer to a power law) when there is no abuse, with a very large number of very small networks (less than five users).

Looking at the reciprocity, there is again a relatively homogeneous distribution for the abuse class, centered around 0.7. For the nonabuse class, a part of the distribution is quite similarly homogeneous (albeit around 0.6), but the large majority of instances have either a 0 or 1 Reciprocity, i.e., only unilateral or bilateral edges, respectively. After verification, the former case corresponds to conversations that come to an abrupt end, whereas the latter is just a normally functioning conversation. Both cases are more likely to happen when few users are involved, which is consistent with our observations regarding the Vertex Count. However, both features are only partially correlated, which shows that the abuse class cannot be reduced only to a question of number of users involved in the conversation.

The closeness seems to have a special role, since its weighted variants constitute their own clusters for the Before (Cluster 110) and After (Cluster 177) graphs. By comparison, the average unweighted Closeness is correlated with many other features as it belongs to the large Cluster 10: this is consistent with our previous observation that certain weighted variants appear to be more informative. Further examination shows that the closeness follows a power law-like distribution in both classes, covering 3 orders of magnitude. However, this heterogeneity is much more marked in the case of the nonabuse class. Concretely, the closeness is generally higher for the abuse class. This means that the average distance between the author of the targeted message and the rest of the graph decreases in case of abuse. This user becomes less peripheral (or more central), and the same goes for the other users of the graph (in average). This fits in quite well with assumptions about how abuse impacts a discussion: an abuser would tend not to be peripheral in a conversation, while we can reasonably assume that the other participants will be piling on and, therefore, be less peripheral themselves.

Most mesoscopic measures are discarded during the feature elimination process. The only remaining ones are the clique count and the coreness, which are also the only ones not related to community structure. Yet, we had considered them as promising in Section IV-B, because they are uncorrelated with the others: it turns out the unique information they convey does not help solving this specific classification task. Inspection reveals that the modularity measure is overwhelmingly close to zero in both classes. This means that our networks generally do not have any community structure, which explains why the related features are not discriminative here.

We also have identified and studied the clusters corresponding to the TFs of the Before, After, and Full feature sets. For matters of space, we do not present them in detail, though, and only discuss our most interesting observations. Certain clusters identified for the All feature set also appear for the other sets: those focusing on the considered type of graph, i.e., Clusters 110 and 118 for Before; 10, 41, 49, and 177 for After, and 9, 10, 41, and 49 for Full. Some of the missing clusters are replaced by semantically close and relatively correlated clusters. For instance, Before has a cluster containing exactly the same measure variants as Cluster 49 (vertex count and betweenness), but for the Before graph. Similarly, Full has a cluster focusing on the weighted closeness, as Cluster 177 does for After. We interpret Clusters 9 and 41 as describing the microscopic aspects of the After graph at the graph and vertex

TABLE III
BEST PERFORMANCES FOR THE BASELINES AND CURRENT FRAMEWORK

Method	Reference	F-measure
Content-based	[17]	76.50
Graph-based	[3]	77.00
Extended graph-based – *All*	This article	**83.89**
Extended graph-based – *All-TF*	This article	82.65

scale, respectively: Before has comparable clusters focusing similarly on the Before graph. Overall, we can say that when focusing on a specific type of graph (by opposition to All), the classifier takes advantage of informationally close clusters, albeit inferior in terms of discriminative power, as they lead to a lower performance.

G. Baseline Comparison

For matters of exhaustiveness, we assess the performance of our framework on a balanced version of our classes, instead of the unbalanced ones used throughout this section. In this setting, the abuse class stays the same, but the nonabuse class is reduced to the size of the abuse one, through sampling. When using these data, we observe a significant performance improvement for all feature sets. In particular, the F-measure values obtained for All and All-TF increase from 83.99 and 82.65 to 88.87 and 87.10, respectively. Further investigations shows that this improvement is mainly due to a decrease in the number of false positives, itself caused by the smaller size of the nonabuse class. Such a balanced situation is unlikely in practice, though.

Finally, we compare the results obtained using our framework with our two baselines (Table III): the content-based approach of [17] and the previous version of our graph-based method [3]. As a reminder, the main differences between the latter and our present framework are that we now extract a directed graph, and use a much larger number of topological measures as classification features. The combination of these two improvements leads to a significant performance increase over our previous effort. As described in Section IV-E, the contribution of edge directions to the overall performance is relatively minor. One could assume that the performance improvement is mainly caused by the major expansion of the feature set, however, this improvement is observed even when only using the TFs identified in Section IV-F. Yet, there are only 10 of them, by comparison to the 75 features used in [3]. So the conclusion here is that both extracting a directed graph and selecting a more appropriate set of features (in particular, topological measures able to handle edge weights) helped improve the performance. More importantly, the performance is greatly improved compared to our content-based approach [17], which is quite representative of the preprocessing and features used in the literature when classifying such data. This is a major result, as it shows that the sole structure of the conversation is enough to efficiently detect abuses, without considering at all the content of the exchanged messages.

V. CONCLUSION

In this paper, we tackle the problem of automatic abuse detection in online communities. We propose to model online conversations through graphs, and to perform the classification task using only graph-based features. The method, while simple, yields good results (up to a 83.89 F-measure), besting the score obtained with a content-based approach [17] and our previous graph-based effort [3]. It completely ignores the content of messages exchanged between users of an online community, which means it is robust to intentional obfuscation of messages by abusive users, as well as unintentional content noise. It is also inherently language independent. One important limitation of our method is the high computational time required to extract the features. However, we show that it can be very significantly reduced by working with a small subset of relevant features, resulting in more than 97% of the original performance for less than 0.01‰ of the processing time. We also show that while our method is originally not designed for real-time abuse detection, the information available at the time the message appears is discriminative enough to do so.

A straightforward extension of our work is to take advantage of both content- and graph-based features, an approach previously applied in other contexts [57]. In our case, they are both based on completely different types of information, so we can assume they are complementary, which could improve the classification performance. At the very least, it will be interesting to combine the features of the Before graph with textual features since that can lead to a system useful for a prediction task. We also consider using a content-based classifier in a completely different ways, during the graph extraction process. Such a classifier could be trained to detect the nature of the interaction between two users, allowing to extract a signed network (negative edge for a hostile exchange, positive otherwise). This additional information is likely to improve the performance of our graph-based classifier.

Finally, part of the future work will focus on applying our proposed approaches to other types of social network corpora. Indeed, chat logs are a special case of communications records that have a very specific structure (entanglement of discussions, near-synchronous communications, various topics in a single flow of discussion, uncertainty about who is the intended recipient of a message...) which do not necessarily appear in other forms of social networks, such as forums or microblogs. For example, since our results have shown that directionality is not the dominant graph construction parameter, we would be interested in evaluating its impact on a type of social media integrating a clear response structure (i.e., a clear identification of who answers whom, such as in a forum like Reddit or a tree-shaped comment section of a news website). In addition, another type of social network corpora might present a more distinct community structure and, therefore, render the meso-scale features we have presented more relevant.

REFERENCES

[1] French Republic. (2004). *Loi n°2004-575 du 21 Juin 2004 Pour la Confiance dans L'économie Numérique—Article 6*. [Online]. Available: https://www.legifrance.gouv.fr/affichTexteArticle.do?idArticle=LEGIARTI000023711900&cidTexte=LEGITEXT000005789847

[2] French Republic. (1982). *Loi n°82-652 du 29 Juillet 1982 sur la Communication Audiovisuelle—Article 93-3*. [Online]. Available: https://www.legifrance.gouv.fr/affichTexteArticle.do?idArticle=LEGIARTI000020740559&cidTexte=LEGITEXT000006068759

[3] E. Papegnies, V. Labatut, R. Dufour, and G. Linarès, "Graph-based features for automatic online abuse detection," in *Proc. Int. Conf. Stat. Lang. Speech Process.* Berlin, Germany: Springer, 2017, pp. 70–81.

[4] E. Spertus, "Smokey: Automatic recognition of hostile messages," in *Proc. 14th Nat. Conf. Artif. Intell. 9th Conf. Innov. Appl. Artif. Intell. (AAAI)*, 1997, pp. 1058–1065.

[5] Y. Chen, Y. Zhou, S. Zhu, and H. Xu, "Detecting offensive language in social media to protect adolescent online safety," in *Proc. IEEE Int. Conf. Privacy, Secur., Risk Trust Int. Conf. Social Comput.*, Sep. 2012, pp. 71–80.

[6] K. Dinakar, R. Reichart, and H. Lieberman, "Modeling the detection of textual cyberbullying," in *Proc. 5th Int. AAAI Conf. Weblogs Social Media/Workshop Social Mobile Web*, 2011, pp. 11–17.

[7] V. S. Chavan and S. S. Shylaja, "Machine learning approach for detection of cyber-aggressive comments by peers on social media network," in *Proc. IEEE Int. Conf. Adv. Comput., Commun. Inform.*, Aug. 2015, pp. 2354–2358.

[8] H. Mubarak, K. Darwish, and W. Magdy, "Abusive language detection on Arabic social media," in *Proc. 1st Workshop Abusive Lang. Online*, 2017, pp. 52–56.

[9] A. H. Razavi, D. Inkpen, S. Uritsky, and S. Matwin, "Offensive language detection using multi-level classification," in *Proc. Can. Conf. Artif. Intell.* Berlin, Germany: Springer, 2010, pp. 16–27.

[10] N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, and N. Bhamidipati, "Hate speech detection with comment embeddings," in *Proc. ACM 24th Int. Conf. World Wide Web*, 2015, pp. 29–30.

[11] J. H. Park and P. Fung, "One-step and two-step classification for abusive language detection on Twitter," in *Proc. 1st Workshop Abusive Lang. Online*, 2017, pp. 41–45.

[12] J. Pavlopoulos, P. Malakasiotis, and I. Androutsopoulos, "Deep learning for user comment moderation," in *Proc. 1st Workshop Abusive Lang. Online*, 2017, pp. 25–35.

[13] H. Hosseini, S. Kannan, B. Zhang, and R. Poovendran. (2017). "Deceiving Google's perspective API built for detecting toxic comments." [Online]. Available: https://arxiv.org/abs/1702.08138

[14] H. Lee and A. Y. Ng, "Spam deobfuscation using a hidden Markov model," in *Proc. 2nd Conf. Email Anti-Spam*, 2005, pp. 1–8.

[15] S. Rojas-Galeano, "On obstructing obscenity obfuscation," *ACM Trans. Web*, vol. 11, no. 2, p. 12, 2017.

[16] D. Yin, Z. Xue, L. Hong, B. D. Davison, A. Kontostathis, and L. Edwards, "Detection of harassment on Web 2.0," in *Proc. Content Anal. WEB*, 2009, pp. 1–7

[17] E. Papegnies, V. Labatut, R. Dufour, and G. Linares, "Impact of content features for automatic online abuse detection," in *Proc. Int. Conf. Comput. Linguistics Intell. Text Process.* Berlin, Germany: Springer, 2017, pp. 404–419.

[18] J. Cheng, C. Danescu-Niculescu-Mizil, and J. Leskovec, "Antisocial behavior in online discussion communities," in *Proc. Int. AAAI Conf. Web Social Media*, 2015, pp. 61–70.

[19] K. Balci and A. A. Salah, "Automatic analysis and identification of verbal aggression and abusive behaviors for online social games," *Comput. Hum. Behav.*, vol. 53, pp. 517–526, Dec. 2015.

[20] P. Mutton, "Inferring and visualizing social networks on Internet relay chat," in *Proc. IEEE 8th Int. Conf. Inf. Vis.*, Jul. 2004, pp. 35–43.

[21] O. I. Osesina, J. P. McIntire, P. R. Havig, E. E. Geiselman, C. Bartley, and M. E. Tudoreanu, "Methods for extracting social network data from chatroom logs," *Proc. SPIE*, vol. 8389, p. 83891H, Jun. 2012. [Online]. Available: https://www.spiedigitallibrary.org/conferenceproceedings-of-spie/8389/83891H/Methods-for-extracting-social-network-data-from-chatroom-logs/10.1117/12.920019.short?SSO=1

[22] A. Gruzd and C. Haythornthwaite, "Automated discovery and analysis of social networks from threaded discussions," in *Proc. Int. Netw. Social Netw. Anal. Conf.*, 2008. [Online]. Available: https://repository.arizona.edu/handle/10150/105081

[23] A. Çamtepe, M. S. Krishnamoorthy, and B. Yener, "A tool for Internet chatroom surveillance," in *Proc. Int. Conf. Intell. Secur. Inform.* Berlin, Germany: Springer, 2004, pp. 252–265.

[24] M. Forestier, J. Velcin, and D. Zighed, "Extracting social networks to understand interaction," in *Proc. Int. Conf. Adv. Social Netw. Anal. Mining*, Jul. 2011, pp. 213–219.

[25] S. Tavassoli, M. Moessner, and K. A. Zweig, "Constructing social networks from semi-structured chat-log data," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining*, Aug. 2014, pp. 146–149.

[26] T. Sinha and I. Rajasingh, "Investigating substructures in goal oriented online communities: Case study of Ubuntu IRC," in *Proc. IEEE Int. Adv. Comput. Conf.*, Feb. 2014, pp. 916–922.

[27] T. Anwar and M. Abulaish, "A social graph based text mining framework for chat log investigation," *Digit. Invest.*, vol. 11, no. 4, pp. 349–362, 2014.

[28] K. Garimella, G. De Francisci Morales, A. Gionis, and M. Mathioudakis, "Quantifying controversy on social media," in *Proc. 9th ACM Int. Conf. Web Search Data Mining*, 2015, pp. 33–42.

[29] D. R. White and F. Harary, "The cohesiveness of blocks in social networks: Node connectivity and conditional density," *Sociol. Methodol. Banner*, vol. 31, no. 1, pp. 305–359, 2001.

[30] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 69, p. 026113, Feb. 2004.

[31] R. D. Luce and A. D. Perry, "A method of matrix analysis of group structure," *Psychometrika*, vol. 14, no. 2, pp. 95–116, 1949.

[32] S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications*, vol. 8. Cambridge, U.K.: Cambridge Univ. Press, 1994,

[33] M. E. J. Newman, "Assortative mixing in networks," *Phys. Rev. Lett.*, vol. 89, no. 20, p. 208701, Oct. 2002.

[34] P. Bonacich, "Factoring and weighting approaches to status scores and clique identification," *J. Math. Sociol.*, vol. 2, no. 1, pp. 113–120, 1972.

[35] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *J. ACM* , vol. 46, no. 5, pp. 604–632, 1999.

[36] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.

[37] P. Bonacich, "Power and centrality: A family of measures," *Amer. J. Sociol.*, vol. 92, no. 5, pp. 1170–1182, 1987.

[38] S. Brin and L. Page, "The anatomy of a large-scale hypertextual Web search engine," *Comput. Netw. ISDN Syst.*, vol. 30, nos. 1–7, pp. 107–117, Apr. 1998.

[39] E. Estrada and J. A. Rodríguez-Velázquez, "Subgraph centrality in complex networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 71, no. 5, p. 056103, 2005.

[40] L. C. Freeman, "A set of measures of centrality based on betweenness," *Sociometry*, vol. 40, no. 1, pp. 35–41, Mar. 1977.

[41] A. Bavelas, "Communication patterns in task-oriented groups," *J. Acoust. Soc. Amer.*, vol. 22, no. 6, pp. 725–730, 1950.

[42] F. Harary, *Graph Theory*. Reading, MA, USA: Addison-Wesley, 1969.

[43] S. B. Seidman, "Network structure and minimum degree," *Social Netw.*, vol. 5, no. 3, pp. 269–287, 1983.

[44] R. Guimerà and L. A. N. Amaral, "Functional cartography of complex metabolic networks," *Nature*, vol. 433, pp. 895–900, Feb. 2005.

[45] V. Labatut, N. Dugué, and A. Perez, "Identifying the community roles of social capitalists in the Twitter network," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining*, Aug. 2014, pp. 371–374.

[46] M. E. Shaw, "Group structure and the behavior of individuals in small groups," *J. Psychol.*, vol. 38, no. 1, pp. 139–149, 1954.

[47] A. Barrat, M. Barthélemy, R. Pastor-Satorras, and A. Vespignani, "The architecture of complex weighted networks," *Proc. Nat. Acad. Sci. USA*, vol. 101, no. 11, pp. 3747–3752, Mar. 2004.

[48] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[49] R. S. Burt, "Structural holes and good ideas1," *Amer. J. Sociol.*, vol. 110, no. 2, pp. 349–399, 2004.

[50] L. C. Freeman, "Centrality in social networks conceptual clarification," *Social Netw.*, vol. 1, no. 3, pp. 215–239, 1979.

[51] P. Bonacich and P. Lloyd, "Eigenvector-like measures of centrality for asymmetric relations," *Social Netw.*, vol. 23, no. 3, pp. 191–201, 2001.

[52] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," *Proc. Nat. Acad. Sci. USA*, vol. 105, no. 4, pp. 1118–1123, 2008.

[53] N. Dugué, V. Labatut, and A. Perez, "A community role approach to assess social capitalists visibility in the Twitter network," *Social Netw. Anal. Mining*, vol. 5, p. 26, Dec. 2015.

[54] G. Csardi and T. Nepusz, "The igraph software package for complex network research," *Inter J. Complex Syst.*, vol. 1695, no. 5, pp. 1–9, 2006. [Online]. Available: http://www.interjournal.org/manuscript_abstract.php?361100992

[55] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.

[56] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, no. 1, pp. 53–65, 1987.

[57] A. Rumshisky *et al.*, "Combining network and language indicators for tracking conflict intensity," in *Proc. Int. Conf. Social Inform.* Berlin, Germany: Springer, 2017, pp. 391–404.