Abusive Language Detection in Online Conversations by Combining Content- and Graph-based Features

Noé Cécillon, Vincent Labatut, Richard Dufour and Georges Linarès

LIA, Avignon University, France

Correspondence*:
Vincent Labatut
{vincent.labatut}@univ-avignon.fr

ABSTRACT

In recent years, online social networks have allowed world-wide users to meet and discuss. As guarantors of these communities, the administrators of these platforms must prevent users from adopting inappropriate behaviors. This verification task, mainly done by humans, is more and more difficult due to the ever growing amount of messages to check. Methods have been proposed to automatize this moderation process, mainly by providing approaches based on the textual content of the exchanged messages. Recent work has also shown that characteristics derived from the structure of conversations, in the form of conversational graphs, can help detecting these abusive messages. In this paper, we propose to take advantage of both sources of information by proposing fusion methods integrating content- and graph-based features. Our experiments on raw chat logs show that the content of the messages, but also of their dynamics within a conversation contain partially complementary information, allowing performance improvements on an abusive message classification task with a final F-measure of 93.26%.

Keywords: Automatic abuse detection, Content analysis, Conversational graph, Online conversations, Social networks

1 INTRODUCTION

Internet widely impacted the way we communicate. Online communities, in particular, have grown to become important places for interpersonal communications. They get more and more attention from companies to advertise their products or from governments interested in monitoring public discourse. Online communities come in various shapes and forms, but they are all exposed to abusive behavior. The definition of what exactly is considered as abuse depends on the community, but generally includes personal attacks, as well as discrimination based on race, religion or sexual orientation.

Abusive behavior is a risk, as it is likely to make important community members leave, therefore endangering the community, and even trigger legal issues in some countries. Moderation consists in detecting users who act abusively, and in taking action against them. Currently this moderation work is mainly a manual process, and since it implies high human and financial costs, companies have a keen interest in its automation. One way of doing so is to consider this task as a classification problem consisting in automatically determining if a user message is abusive or not.

A number of works have tackled this problem, or related ones, in the literature. Most of them focus only on the content of the targeted message to detect abuse or similar properties. For instance, Spertus (1997) applies this principle to detect hostility, Dinakar et al. (2011) for cyberbullying, and Chen et al. (2012) for offensive language. These approaches rely on a mix of standard NLP features and manually crafted application-specific resources (e.g. linguistic rules). We also proposed a content-based method (Papegnies et al., 2017a) using a wide array of language features (Bag-of-Words, tf-idf scores, sentiment scores). Other approaches are more machine learning intensive, but require larger amounts of data. Recently, Wulczyn et al. (2017) created three datasets containing individual messages collected from Wikipedia discussion pages, annotated for toxicity, personal attacks and aggression, respectively. They have been leveraged in recent works to train Recursive Neural Network operating on word embeddings and character n-gram features (Pavlopoulos et al., 2017; Mishra et al., 2018). However, the quality of these direct content-based approaches is very often related to the training data used to learn abuse detection models. In the case of online social networks, the great variety of users, including very different language registers, spelling mistakes, as well as intentional users obfuscation, makes it almost impossible to have models robust enough to be applied in all cases. (Hosseini et al., 2017) have then shown that it is very easy to bypass automatic toxic comment detection systems by making the abusive content difficult to detect (intentional spelling mistakes, uncommon negatives...).
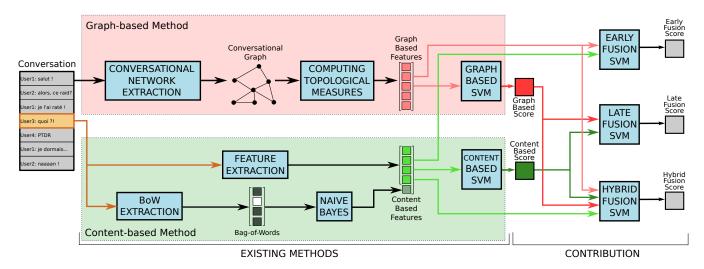
Because the reactions of other users to an abuse case are completely beyond the abuser's control, some authors consider the content of messages occurring *around* the targeted message, instead of focusing only on the targeted message itself. For instance, (Yin et al., 2009) use features derived from the sentences neighboring a given message to detect harassment on the Web. (Balci and Salah, 2015) take advantage of user features such as the gender, the number of in-game friends or the number of daily logins to detect abuse in the community of an online game. In our previous work (Papegnies et al., 2019), we proposed a radically different method that completely ignores the textual content of the messages, and relies only on a graph-based modeling of the conversation. This is the only graph-based approach ignoring the linguistic content proposed in the context of abusive messages detection. Our conversational network extraction process is inspired from other works leveraging such graphs for other purposes: chat logs (Mutton, 2004) or online forums (Forestier et al., 2011) interaction modeling, user group detection (Camtepe et al., 2004). Additional references on abusive message detection and conversational network modeling can be found in (Papegnies et al., 2019).

In this paper, based on the assumption that the interactions between users and the content of the exchanged messages convey different information, we propose a new method to perform abuse detection while leveraging both sources. For this purpose, we take advantage of the content-(Papegnies et al., 2017b) and graph-based (Papegnies et al., 2019) methods that we previously developed. We propose three different ways to combine them, and compare their performance on a corpus of chat logs originating from the community of a French multiplayer online game. We then perform a feature study, finding the most informative ones and discussing their role. Our contribution is twofold: the exploration of fusion methods, and more importantly the identification of discriminative features for this problem.

The rest of this article is organized as follows. In Section 2, we describe the methods and strategies used in this work. In Section 3 we present our dataset, the experimental setup we use for this classification task, and the performances we obtained. Finally, we summarize our contributions in Section 4 and present some perspectives for this work.

2 METHODS

In this section, we summarize the content-based method from (Papegnies et al., 2017b) (Section 2.1) and the graph-based method from (Papegnies et al., 2019) (Section 2.2). We then present the fusion method proposed in this paper, aiming at taking advantage of both sources of information (Section 2.3). Figure 1 shows the whole process, and is discussed through this section.

Figure 1. Representation of our processing pipeline. *Existing methods* refers to our previous work described in (Papegnies et al., 2017b) (content-based method) and (Papegnies et al., 2019) (graph-based method), whereas the contribution presented in this article appears on the right side (fusion strategies). Figure available at 10.6084/m9.figshare.7442273 under CC-BY license.

2.1 Content-Based Method

This method corresponds to the bottom-left part of Figure 1 (in green). It consists in extracting certain features from the content of each considered message, and to train a Support Vector Machine (SVM) classifier to distinguish abusive (*Abuse* class) and non-abusive (*Non-abuse* class) messages (Papegnies et al., 2017b). These features are quite standard in Natural Language Processing (NLP), so we only describe them briefly here.
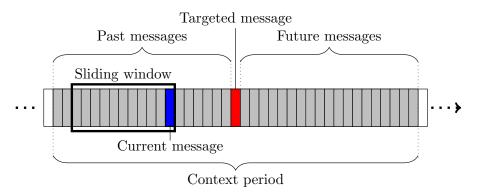
We use a number of *morphological features*. We use the message length, average word length, and maximal word length, all expressed in number of characters. We count the number of unique characters in the message. We distinguish between six classes of characters (letters, digits, punctuation, spaces, and others) and compute two features for each one: number of occurrences, and proportion of characters in the message. We proceed similarly with capital letters. Abusive messages often contain a lot of copy/paste. To deal with such redundancy, we apply the Lempel–Ziv–Welch (LZW) compression algorithm (Batista and Meira, 2004) to the message and take the ratio of its raw to compress lengths, expressed in characters. Abusive messages also often contain extra-long words, which can be identified by collapsing the message: extra occurrences of letters repeated more than two times consecutively are removed. For instance, "looooooool" would be collapsed to "lool". We compute the difference between the raw and collapsed message lengths.

We also use *language features*. We count the number of words, unique words and bad words in the message. For the latter, we use a predefined list of insults and symbols considered as abusive, and we also count them in the collapsed message. We compute two overall tf–idf scores corresponding to the sums of

the standard $tf-idf$ scores of each individual word in the message. One is processed relatively to the *Abuse* class, and the other to the *Non-abuse* class. We proceed similarly with the collapsed message. Finally, we lower-case the text and strip punctuation, in order to represent the message as a basic Bag-of-Words (BoW). We then train a Naive Bayes classifier to detect abuse using this sparse binary vector (as represented in the very bottom part of Figure 1). The output of this simple classifier is then used as an input feature for the SVM classifier.
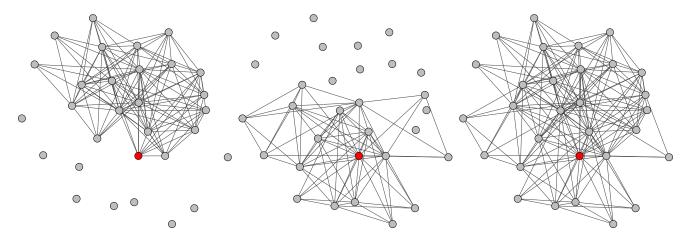
2.2   Graph-Based Method

This method corresponds to the top-left part of Figure 1 (in red). It completely ignores the content of the messages, and only focuses on the dynamics of the conversation, based on the interactions between its participants (Papegnies et al., 2019). It is three-stepped: 1) extracting a conversational graph based on the considered message as well as the messages preceding and/or following it; 2) computing the topological measures of this graph to characterize its structure; and 3) using these values as features to train an SVM to distinguish between abusive and non-abusive messages. The vertices of the graph model the participants of the conversation, whereas its weighted edges represent how intensely they communicate.

Figure 2. Illustration of the main concepts used during network extraction (see text for details). Figure available at 10.6084/m9.figshare.7442273 under CC-BY license.

The graph extraction is based on a number of concepts illustrated in Figure 2, in which each rectangle represents a message. The extraction process is restricted to a so-called *context period*, i.e. a sub-sequence of messages including the message of interest, itself called *targeted message* and represented in red in Figure 2. Each participant posting at least one message during this period is modeled by a vertex in the produced conversational graph. A mobile window is slid over the whole period, one message at a time. At each step, the network is updated either by creating new links, or by updating the weights of existing ones. This *sliding window* has a fixed length expressed in number of messages, which is derived from ergonomic constraints relative to the online conversation platform studied in Section 3. It allows focusing on a smaller part of the context period. At a given time, the last message of the window (in blue in Figure 2) is called *current message* and its author *current author*. The weight update method assumes that the current message is aimed at the authors of the other messages present in the window, and therefore connects the current author to them (or strengthens their weights if the edge already exists). It also takes chronology into account by favoring the most recent authors in the window. Three different variants of the conversational network are extracted for one given targeted message: the *Before* network is based on the messages posted before the targeted message, the *After* network on those posted after, and the *Full* network on the whole

context period. Figure 3 shows an example of such networks obtained for a message of the corpus described in Section 3.1.

Figure 3. 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. Figure available at 10.6084/m9.figshare.7442273 under CC-BY license.

Once the conversational networks have been extracted, they must be described through numeric values in order to feed the SVM classifier. This is done through a selection of standard topological measures allowing to describe a graph in a number of distinct ways, focusing on different scales and scopes. The *scale* denotes the nature of the characterized entity. In this work, the individual vertex and the whole graph are considered. When considering a single vertex, the measure focuses on the *targeted author* (i.e. the author of the targeted message). The *scope* can be either micro-, meso- or macroscopic: it corresponds to the amount of information considered by the measure. For instance, the graph density is microscopic, the modularity is mesoscopic, and the diameter is macroscopic. All these measures are computed for each graph, and allow describing the conversation surrounding the message of interest. The SVM is then trained using these values as features. In this work, we use exactly the same measures as in (Papegnies et al., 2019).

2.3 Fusion

We now propose a new method seeking to take advantage of both previously described ones. It is based on the assumption that the content- and graph-based features convey different information. Therefore, they could be complementary, and their combination could improve the classification performance. We experiment with three different fusion strategies, which are represented in the right-hand part of Figure 1.

The first strategy follows the principle of *Early Fusion*. It consists in constituting a global feature set containing all content- and graph-based features from Sections 2.1 and 2.2, then training a SVM directly using these features. The rationale here is that the classifier has access to the whole raw data, and must determine which part is relevant to the problem at hand.

The second strategy is *Late Fusion*, and we proceed in two steps. First, we apply separately both methods described in Sections 2.1 and 2.2, in order to obtain two scores corresponding to the output probability of each message to be abusive given by the content- and graph-based methods, respectively. Second, we fetch these two scores to a third SVM, trained to determine if a message is abusive or not. This approach relies on the assumption that these scores contain all the information the final classifier needs, and not the noise present in the raw features.

Finally, the third fusion strategy can be considered as *Hybrid Fusion*, as it seeks to combine both previous proposed ones. We create a feature set containing the content- and graph-based features, like with *Early Fusion*, but also both scores used in *Late Fusion*. This whole set is used to train a new SVM. The idea is to check whether the scores do not convey certain useful information present in the raw features, in which case combining scores and features should lead to better results.

3 EXPERIMENTS

In this section, we first describe our dataset and the experimental protocol followed in our experiments (Section 3.1). We then present and discuss our results, in terms of classification performance (Sections 3.2) and feature selection (Section 3.3).

3.1 Experimental protocol

The dataset is the same as in our previous publications (Papegnies et al., 2017b, 2019). It is a proprietary database containing 4,029,343 messages in French, exchanged on the in-game chat of *SpaceOrigin*[1], a Massively Multiplayer Online Role-Playing Game (MMORPG). Among them, 779 have been flagged as being abusive by at least one user in the game, and confirmed as such by a human moderator. They constitute what we call the *Abuse* class. Some inconsistencies in the database prevent us from retrieving the context of certain messages, which we remove from the set. After this cleaning, the *Abuse* class contains 655 messages. In order to keep a balanced dataset, we further extract the same number of messages at random from the ones that have not been flagged as abusive. This constitutes our *Non-abuse* class. Each message, whatever its class, is associated to its surrounding context (i.e. messages posted in the same thread).

The graph extraction method used to produce the graph-based features requires to set certain parameters. We use the values matching the best performance, obtained during the greedy search of the parameter space performed in (Papegnies et al., 2019). In particular, regarding the two most important parameters (see Section 2.2), we fix the *context period* size to 1,350 messages and the *sliding window* length to 10 messages. Implementation-wise, we use the iGraph library (Csardi and Nepusz, 2006) to extract the conversational networks and process the corresponding features. We use the Sklearn toolkit (Pedregosa et al., 2011) to get the text-based features. We use the SVM classifier implemented in Sklearn under the name SVC (C-Support Vector Classification). Because of the relatively small dataset, we set-up our experiments using a 10-fold cross-validation. Each fold is balanced between the *Abuse* and *Non-abuse* classes, 70% of the dataset being used for training and 30% for testing.

3.2 Classification Performance

Table 1 presents the Precision, Recall and F-measure scores obtained on the *Abuse* class, for both baselines (*Content-based* (Papegnies et al., 2017b) and *Graph-based* (Papegnies et al., 2019)) and all three proposed fusion strategies (*Early Fusion*, *Late Fusion* and *Hybrid Fusion*). It also shows the number of features used to perform the classification, the time required to compute the features and perform the cross validation (*Total Runtime*) and to compute one message in average (*Average Runtime*). Note that *Late Fusion* has only 2 direct inputs (content- and graph-based SVMs), but these in turn have their own inputs, which explains the values displayed in the table.

---

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

Our first observation is that we get higher F-measure values compared to both baselines when performing the fusion, independently from the fusion strategy. This confirms what we expected, i.e. that the information encoded in the interactions between the users differs from the information conveyed by the content of the messages they exchange. Moreover, this shows that both sources are at least partly complementary, since the performance increases when merging them. On a side note, the correlation between the score of the graph- and content-based classifiers is 0.56, which is consistent with these observations.

Next, when comparing the fusion strategies, it appears that *Late Fusion* performs better than the others, with an F-measure of 93.26. This is a little bit surprising: we were expecting to get superior results from the *Early Fusion*, which has direct access to a much larger number of *raw* features (488). By comparison, the *Late Fusion* only gets 2 features, which are themselves the outputs of two other classifiers. This means that the *Content-Based* and *Graph-Based* classifiers do a good work in summarizing their inputs, without loosing much of the information necessary to efficiently perform the classification task. Moreover, we assume that the *Early Fusion* classifier struggles to estimate an appropriate model when dealing with such a large number of features, whereas the *Late Fusion* one benefits from the pre-processing performed by its two predecessors, which act as if reducing the dimensionality of the data. This seems to be confirmed by the results of the *Hybrid Fusion*, which produces better results than the *Early Fusion*, but is still below the *Late Fusion*. This point could be explored by switching to classification algorithm less sensitive to the number of features. Alternatively, when considering the three SVMs used for the *Late Fusion*, one could see a simpler form of a very basic Multilayer Perceptron, in which each neuron has been trained separately (without system-wide backpropagation). This could indicate that using a regular Multilayer Perceptron directly on the raw features could lead to improved results, especially if enough training data is available.

Regarding runtime, the graph-based approach takes more than 8 hours to run for the whole corpus, mainly because of the feature computation step. This is due to the number of features, and to the compute-intensive nature of some of them. The content-based approach is much faster, with a total runtime of less than 1 minute, for the exact opposite reasons. Fusion methods require to compute both content- and graph-based features, so they have the longest runtime.

Table 1. Comparison of the performances obtained with the methods (*Content-based*, *Graph-based*, *Fusion*) and their subsets of *Top Features* (TF). The total runtime is expressed as *h:min:s*. See text for details.

Method	Number of features	Total Runtime	Average Runtime	Precision	Recall	F-measure
Content-Based	29	0:52	0.02s	78.59	83.61	81.02
Content-Based TF	3	0:21	0.01s	75.82	82.57	79.05
Graph-Based	459	8:19:10	7.56s	90.21	87.63	88.90
Graph-Based TF	10	14:22	0.03s	88.72	84.87	86.75
Early Fusion	488	8:26:41	7.68s	91.25	89.45	90.34
Early Fusion TF	4	11:29	0.17s	89.09	87.12	88.09
Late Fusion	488 (2)	8:23:57	7.64s	94.10	92.43	93.26
Late Fusion TF	13	15:42	0.24s	91.64	89.97	90.80
Hybrid Fusion	490	8:27:01	7.68s	91.96	90.48	91.22
Hybrid Fusion TF	4	16:57	0.26s	90.74	89.00	89.86

3.3   Feature Study

We now want to identify the most discriminative features for all three fusion strategies. We apply an iterative method based on the *Sklearn* toolkit, which allows us to fit a linear kernel SVM to the dataset

and provide a ranking of the input features reflecting their importance in the classification process. Using this ranking, we identify the least discriminant feature, remove it from the dataset, and train a new model with the remaining features. The impact of this deletion is measured by the performance difference, in terms of F-measure. We reiterate this process until only one feature remains. We call *Top Features* (TF) the minimal subset of features allowing to reach 97% of the original performance (when considering the complete feature set).

We apply this process to both baselines and all three fusion strategies. We then perform a classification using only their respective TF. The results are presented in Table 1. Note that the *Late Fusion TF* performance is obtained using the scores produced by the SVMs trained on *Content-based TF* and *Graph-based TF*. These are also used as features when computing the TF for *Hybrid Fusion TF* (together with the raw content- and graph-based features). In terms of classification performance, by construction, the methods are ranked exactly like when considering all available features.

The *Top Features* obtained for each method are listed in Table 2. The last 4 columns precise which variants of the graph-based features are concerned. Indeed, as explained in Section 2.2, most of these topological measures can handle/ignore edge weights and/or edge directions, can be vertex- or graph-focused, and can be computed for each of the three types of networks (*Before*, *After* and *Full*).

There are three *Content-Based TF*. The first is the *Naive Bayes* prediction, which is not surprising as it comes from a fully fledged classifier processing BoWs. The second is the $tf\text{-}idf$ *score* computed over the *Abuse* class, which shows that considering term frequencies indeed improve the classification performance. The third is the *Capital Ratio* (proportion of capital letters in the comment), which is likely to be caused by abusive message tending to be shouted, and therefore written in capitals. The *Graph-Based TF* are discussed in depth in our previous article (Papegnies et al., 2019). To summarize, the most important features help detecting changes in the direct neighborhood of the targeted author (Coreness, Strength), in the average node centrality at the level of the whole graph in terms of distance (Closeness), and in the general reciprocity of exchanges between users (Reciprocity).

We obtain 4 features for *Early Fusion TF*. One is the *Naive Bayes* feature (content-based), and the other three are topological measures (graph-based features). Two of the latter correspond to the Coreness of the targeted author, computed for the *Before* and *After* graphs. The third topological measure is his/her Eccentricity. This reflects important changes in the interactions around the targeted author. It is likely caused by angry users piling up on the abusive user after he has posted some inflammatory remark. For *Hybrid Fusion TF*, we also get 4 features, but those include in first place both SVM outputs from the content- and graph-based classifiers. Those are completed by 2 graph-based features, including Strength (also found in the *Graph-based* and *Late Fusion TF*) and Coreness (also found in the *Graph-based*, *Early Fusion* and *Late Fusion TF*).

Besides a better understanding of the dataset and classification process, one interesting use of the TF is that they can allow decreasing the computational cost of the classification. In our case, this is true for all methods: we can retain 97% of the performance while using only a handful of features instead of hundreds. For instance, with the *Late Fusion TF*, we need only 3% of the total *Late Fusion* runtime.

4 CONCLUSION AND PERSPECTIVES

In this article, we tackle the problem of automatic abuse detection in online communities. We take advantage of the methods that we previously developed to leverage message content (Papegnies et al., 2017a) and interactions between users (Papegnies et al., 2019), and create a new method using both types

Table 2. Top features obtained for our 5 methods. The letters in the *Graph* column stand for *Before* (B), *After* (A) and *Full* (F). Those in the *Weights* and *Directions* columns stand for: *Unweighted* or *Undirected* (U), *Weighted* (W), *Directed* (D), *Incoming* (I) and *Outgoing* (O). Those in the *Scale* column mean *Graph-scale* (G) or *Vertex-scale* (N).

Method	Top Features	Graph	Weights	Directions	Scale
Content-Based	Naive Bayes	–	–	–	–
	$tf{-}idf$ Abuse Score	–	–	–	–
	Character Capital Ratio	–	–	–	–
Graph-Based	Coreness Score	F	–	I	G
	PageRank Centrality	A	U	D	N
	Strength Centrality	F	W	O	N
	Vertex Count	F	–	–	G
	Closeness Centrality	B	W	O	G
	Closeness Centrality	B	W	O	N
	Authority Score	B	W	D	G
	Hub Score	B	U	D	N
	Reciprocity	A	–	D	G
	Closeness Centrality	A	W	U	N
Early Fusion	Coreness Score	A	–	O	G
	Coreness Score	B	–	I	G
	Eccentricity	B	–	I	G
	Naive Bayes	–	–	–	–
Late Fusion	*Content-Based* TF ∪ *Graph-Based* TF	–	–	–	–
Hybrid Fusion	Graph-based output	–	–	–	–
	Content-based output	–	–	–	–
	Strength Centrality	A	W	O	N
	Coreness Score	B	–	I	G

of information simultaneously. We show that the features extracted from our content- and graph-based approaches are complementary, and that combining them allows to sensibly improve the results up to 93.26 (F-measure). One limitation of our method is the computational time required to extract certain features. However, we show that using only a small subset of relevant features allows to dramatically reduce the processing time (down to 3%) while keeping more than 97% of the original performance.

Another limitation of our work is the small size of our dataset. We must find some other corpora to test our methods at a much higher scale. However, all the available datasets are composed of isolated messages, when we need threads to make the most of our approach. A solution could be to start from datasets such as the Wikipedia-based corpus proposed by Wulczyn et al. (2017), and complete them by reconstructing the original conversations containing the annotated messages. This could also be the opportunity to test our methods on an other language than French. Our content-based method may be impacted by this change, but this should not be the case for the graph-based method, as it is independent from the content (and therefore the language). Besides language, a different online community is likely to behave differently from the one we studied before. In particular, its members could react differently differently to abuse. The Wikipedia dataset would therefore allow assessing how such cultural differences affect our classifiers, and identifying which observations made for Space Origin still apply to Wikipedia.

REFERENCES

Balci, K. and Salah, A. A. (2015). Automatic analysis and identification of verbal aggression and abusive behaviors for online social games. *Computers in Human Behavior* 53, 517–526. doi:10.1016/j.chb.2014.10.025

Batista, L. V. and Meira, M. M. (2004). Texture classification using the lempel-ziv-welch algorithm. In *Brazilian Symposium on Artificial Intelligence*. 444–453. doi:10.1007/978-3-540-28645-5_45

Camtepe, A., Krishnamoorthy, M. S., and Yener, B. (2004). A tool for Internet chatroom surveillance. In *International Conference on Intelligence and Security Informatics* (Berlin, DE: Springer), vol. 3073 of *Lecture Notes in Computer Science*, 252–265. doi:10.1007/978-3-540-25952-7_19

Chen, Y., Zhou, Y., Zhu, S., and Xu, H. (2012). Detecting offensive language in social media to protect adolescent online safety. In *International Conference on Privacy, Security, Risk and Trust and International Conference on Social Computing* (IEEE), 71–80. doi:10.1109/SocialCom-PASSAT.2012. 55

Csardi, G. and Nepusz, T. (2006). The igraph software package for complex network research. *InterJournal* 1695

Dinakar, K., Reichart, R., and Lieberman, H. (2011). Modeling the detection of textual cyberbullying. In *5th International AAAI Conference on Weblogs and Social Media / Workshop on the Social Mobile Web* (AAAI), 11–17

Forestier, M., Velcin, J., and Zighed, D. (2011). Extracting social networks to understand interaction. In *International Conference on Advances in Social Networks Analysis and Mining* (IEEE), 213–219. doi:10.1109/ASONAM.2011.64

Hosseini, H., Kannan, S., Zhang, B., and Poovendran, R. (2017). Deceiving google's perspective api built for detecting toxic comments. *preprint arXiv:1702.08138*

Mishra, P., Yannakoudakis, H., and Shutova, E. (2018). Neural character-based composition models for abuse detection. *CoRR* abs/1809.00378

Mutton, P. (2004). Inferring and visualizing social networks on Internet Relay Chat. In *8th International Conference on Information Visualisation* (IEEE), 35–43. doi:10.1109/IV.2004.1320122

Papegnies, E., Labatut, V., Dufour, R., and Linares, G. (2017a). Graph-based features for automatic online abuse detection. In *International Conference on Statistical Language and Speech Processing* (Berlin, DE: Springer), vol. 10583 of *Lecture Notes in Computer Science*, 70–81. doi:10.1007/978-3-319-68456-7_6

Papegnies, E., Labatut, V., Dufour, R., and Linares, G. (2017b). Impact of content features for automatic online abuse detection. In *International Conference on Computational Linguistics and Intelligent Text Processing* (Berlin, DE: Springer), vol. 10762 of *Lecture Notes in Computer Science*, 404–419. doi:10.1007/978-3-319-77116-8_30

Papegnies, E., Labatut, V., Dufour, R., and Linares, G. (2019). Conversational networks for automatic online moderation. In *IEEE Transactions on Computational Social Systems* (IEEE), 38–55. doi:10.1109/TCSS.2018.2887240

Pavlopoulos, J., Malakasiotis, P., and Androutsopoulos, I. (2017). Deep learning for user comment moderation. In *1st Workshop on Abusive Language Online* (ACL), 25–35

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12, 2825–2830

Spertus, E. (1997). Smokey: Automatic recognition of hostile messages. In *14th National Conference on Artificial Intelligence and 9th Conference on Innovative Applications of Artificial Intelligence* (AAAI), 1058–1065

Wulczyn, E., Thain, N., and Dixon, L. (2017). Ex Machina: Personal attacks seen at scale. In *26th International Conference on World Wide Web*. 1391–1399. doi:10.1145/3038912.3052591

Yin, D., Xue, Z., Hong, L., Davison, B. D., Kontostathis, A., and Edwards, L. (2009). Detection of harassment on Web 2.0. In *WWW Workshop: Content Analysis in the Web 2.0*. 1–7