Shielding Google’s language toxicity model against adversarial attacks

Nestor Rodriguez* Sergio Rojas–Galeano**

Abstract

Background. Lack of moderation in online communities enables participants to incur in personal aggression, harassment or cyberbullying, issues that have been accentuated by extremist radicalisation in the contemporary post-truth politics scenario. This kind of hostility is usually expressed by means of toxic language, profanity or abusive statements. Recently Google has developed a machine-learning-based toxicity model in an attempt to assess the hostility of a comment; unfortunately, it has been suggested that said model can be deceived by adversarial attacks that manipulate the text sequence of the comment.

Methods/Results. In this paper we firstly characterise such adversarial attacks as using obfuscation and polarity transformations. The former deceives by corrupting toxic trigger content with typographic edits, whereas the latter deceives by grammatical negation of the toxic content. Then, we propose a two-stage approach to counter-attack these anomalies, building upon a recently proposed text deobfuscation method and the toxicity scoring model. Lastly, we conducted an experiment with approximately 24000 distorted comments, showing how in this way it is feasible to restore toxicity of the adversarial variants, while incurring roughly on a twofold increase in processing time.

Conclusions. Even though novel adversary challenges would keep coming up derived from the versatile nature of written language, we anticipate that techniques combining machine learning and text pattern recognition methods, each one targeting different layers of linguistic features, would be needed to achieve robust detection of toxic language, thus fostering aggression-free digital interaction.

*Independent researcher, Munich, Germany
**Universidad Distrital FJC, School of Engineering, Bogotá, Colombia
Correspondance email: srojas@udistrital.edu.co
1 Introduction

Grecian agora was the public place where citizens in ancient times gathered to debate current affairs and exercise rhetoric as a way to persuade audiences to follow a proposal for action. Nowadays digital media such as social networks platforms, originally conceived as simple virtual cork boards to exchange information among friends, have evolved to become contemporary agoras, where any person with an Internet-connected device may express their opinions and debate them openly and freely.

Unfortunately, not only the medium but also the discourse have changed, and rhetoric arguments are now frequently based on emotion rather than reason, yielding discussions intended to ridicule, distort or confuse other’s opinion, avoiding factual-based debate in favour of opinion manipulation by means of fake news, libel and personal or social group hostility, an scenario now commonly-referred as post-truth politics [4].

Emotion-guided arguments may lead easily to radicalism in political, religious, ethnic, sport or minorities views, which in turn may result in comments coloured with personal aggression, harassment or cyberbullying [7,8,12]. This kind of hostility is becoming a cause of concern in online communities; hence, automatic moderation tools are needed to prevent caustic behaviour within the steadily increasing massive amounts of daily discussions generated in social media. In this direction, Google Counter-Abuse Technology Team has launched Perspective, a tool to identify toxicity of a written comment based on crowd-sourcing and machine learning models trained on large datasets of toxic conversations, as an attempt to provide safer places for online discussions [17].

Despite the remarkable efficacy of this tool to identify high-calibre language in diverse hot topics such as US Presidential election, Brexit and climate change, it has been suggested recently that its detection mechanism can be heavily defeated using adversarial strategies that corrupt the input text sequence with typographic or polarity manipulation, to such a degree that becomes unrecognisable to the trained model but remains readable by the human eye. For example, Hosseini et al. [6] has shown that the insulting statement “They are liberal idiots who are uneducated” (toxicity: 90%), becomes a mild comment when written as “They are liberal i.d.iots who are un.educated” (toxicity: 15%). Similarly, the rude sentence “It’s stupid and wrong” (toxicity: 89%), remains rude even if negated: “It’s not stupid and wrong” (toxicity: 83%). Since the space of possible sequence variations ob-
tainable with these kind of attacks is combinatorial, it would not be feasible to train a model even with large amounts of available examples.

In this paper a counter–attack strategy is devised; it consists of firstly pre–processing the input with a recently proposed text deobfuscation method [14] so as to transform it back into its original representation, and subsequently feeding this corrected text to the toxic scoring model. Our results indicate that this approach is able to effectively restore the intended toxic score of the corrupted text given by Perspective, contributing so to shield Google toxicity model against these sort of adversarial attacks.

2 Materials and Methods

2.1 Google toxicity model

The Google Perspective (GP) model aims to score toxicity of harassment comments in online platforms on a scale from 0 (“healthy”) to 1 (“very toxic”). The model was built based on large–scale datasets of abusive comments, using crowd–sourced annotations to train machine learning classifiers (logistic regression and neural networks, with bag–of–words features, see [17] for details). Toxic content is defined as “rude, disrespectful, or unreasonable comment that is likely to make you leave a discussion” [1].

2.2 Adversarial attacks

Within the machine learning community, an adversarial framework refers to inputs deliberately designed to manipulate the expected behaviour of a prediction model [9, 15]. The adversary usually picks data from distributions different to those assumed when training the model, thus defeating its prediction capabilities. The attacks usually consist of corrupted features or distorted inputs. In our problem of interest, two adversarial attacks on the GP toxic model have been recently suggested [6], which we formally characterise as follows.

Obfuscation attack. In this attack the adversary modifies the character sequence of those words conveying most of the toxic content within the input comment. The modification takes advantage of the robustness of the human visual system to recognise corrupted variants of text. Such edits can be: misspellings or symbol substitution, letter repetition, fake punctuation (in-
serting dots, commas or blanks within letters in the words). As a result, the input text becomes disguised from its original character codification. These kind of obfuscations has been identified as homoglyph substitution and bogus segmentation anomalies [14]. Using these stratagems, attacks have been reported to effectively reduce the toxicity score of an aggressive comment down to a benign level [6].

**Polarity attack.** In this attack the adversary attempts to obtain high toxicities for inoffensive comments that however exhibit profanity content. This is achieved by simply negating toxic terms that effectively swaps the polarity of the comment to its opposite meaning or emotional intent. For example, if the toxic word is an adjective, the polarity change is achieved by inserting the particle “not” before the word. The modified comment still contains the character sequence of the toxic term, misleading the model to identify it as aggressive even though its deceiving neutrality may originate from idiomatically uncommon or odd–looking sentences. This kind of vulnerability was also exploited effectively in [6] to outwit the toxic model. We remark that the adversarial intensity can be worsen by combining the two attacks within the same input, i.e. negating obfuscated versions of toxicity terms.

### 2.3 Dataset preparation

The GP website [1] provides a sample of comments gathered from online surveys on three delicate topics: US Election (45 comments), Brexit (61) and Climate change (49). The comments’ text along with scores obtained with the GP toxicity model are given. So we collected those with toxicity scores higher than 60%, obtaining a subset of 24 comments that were labeled as \{use101,...,use110, brex01,...,brex07, clic01,...,clic07\}. The complete list of comments is provided as supplementary information [Appendix A. Original comments].

We subsequently prepared datasets conveying the two types of adversarial anomalies, namely obfuscation attack and polarity attack. In the first attack, 1000 variations of each comment were randomly generated by isolating the toxic terms of the comment in order to obfuscate them with a number of edits including homoglyph substitution, bogus segmentation and letter repetition (using the same scheme as [14]). The number of edits applied to each term was controlled with a corruption rate \(p \in \{50%,99%\}\). Thus, for each character in the term a uniformly-distributed random number
Table 1. An excerpt of adversarial examples from the datasets.

| Topic               | Attack         | ID            | Text sequence                                      |
|---------------------|----------------|---------------|----------------------------------------------------|
| US Election         | None (original)| use101        | If they voted for Hilary they are idiots           |
|                     | obfuscation-50 | use101-098    | If they voted for Hilary they are !d10nts          |
|                     | obfuscation-99 | use101-295    | If they voted for Hilary they are i;;d-lot,5        |
|                     | polarity       | use101’       | If they voted for Hilary they are NOT idiots        |
| Brexit              | None (original)| brex03        | They are stupid and ignorant with no class         |
|                     | obfuscation-50 | brex03-381    | They are st.VPId and ignor@Nt with no class         |
|                     | obfuscation-99 | brex03-120    | They are s7uuupi-d,, and Ig,Noooooora.u t with no class |
|                     | polarity       | brex03’       | They are NOT stupid and NOT ignorant with no class  |
| Climate change      | None (original)| clic01        | They have their heads up their ass                  |
|                     | obfuscation-50 | clic01-688    | They have their heads up their AS5                  |
|                     | obfuscation-99 | clic01-512    | They have their heads up their a**$s               |
|                     | polarity       | clic01’       | They have their heads NOT up their ass              |

$r \sim \mathcal{U}(0, 1)$ was sampled; if $r < p$ the character was kept unchanged, otherwise one of these edits was applied (with their respective probabilities): an homoglyph substitution (60%), a bogus segmentation (30%) or a fake letter repetition (10%). In this way, we obtained two datasets, obfuscation-50 and obfuscation-99 consisting of 24,000 obfuscated toxic comments each (see examples in Table 1). With respect to the homoglyph edits utilised in the obfuscation attack, we defined substitution lists encompassing a subset of the ASCII encoding set, as depicted in Fig. 1. We note in passing that these lists can be customised to further include additional homoglyphs from extended character encodings (e.g. letters with diacritical marks).

Now, the second dataset (polarity) was obtained by inserting negation predicates within each comment. Thus, the size of this dataset is 24 negated comments. Although possible, we refrained from generating obfuscated variants of these comments as the aim of this attack was to investigate the effect of the polarity change achieved by the negated predicates alone. Table 1 also shows examples for this attack. The complete datasets are provided as supplementary files (see Appendix B. Datasets and experiment source code).

On the other hand, the list of toxic terms was build as follows. For each comment in the original dataset, we scanned every single word with the toxicity model; those scoring higher than 50% were added to the list. Regarding the negated predicates, such list was built by preceding each word within the toxic term list, with the particle not if it was an adjective or noun, or do not if it was a verb. The resulting lists are shown in Table 2.
Figure 1. Homoglyph substitution lists. Each list is indexed by the characters shown in the inner orange ring, corresponding to the letters of the English alphabet. The sets of their corresponding homoglyphs used in this study are shown in the outer blue ring. The blank character “.” indexes the lists of bogus segmentators.

Table 2. The lists of toxic terms and their negated predicates.

| Toxic term  | Negated predicate   | Toxic term  | Negated predicate   |
|-------------|---------------------|-------------|---------------------|
| idiots      | not idiots          | awful       | not awful           |
| stupid      | not stupid          | in hell     | not in hell         |
| the worst   | not the worst       | morons      | not morons          |
| screw you   | don’t screw you     | terrible    | not terrible        |
| suck        | don’t suck          | a shame     | not a shame         |
| stupidity   | no stupidity        | racist      | not racist          |
| sexist      | not sexist          | supremacists| not supremacists    |
| assholes    | not assholes        | ignorant    | not ignorant        |
| a moron     | not a moron         | nationalists| not nationalists     |
| rubbish     | not rubbish         | up their ass| not up their ass    |
| uneducated  | not uneducated      | an idiot    | not an idiot        |
| an abortion | not an abortion     |             |                     |
2.4 Disarming the adversarial attacks

The method we propose to address the adversarial attacks is illustrated in Fig. 2; it basically consist of filtering the corrupted comment prior to feeding the toxicity model, by means of the deobfuscation approach introduced in [14]. Such normalisation filter requires a list of target vocables written in plain English, that are to be searched within the input text. The approach then uses an approximate string matching algorithm to find occurrences of the vocables, either verbatim copies or variants resulting from the homoglyph substitution, bogus segmentation or symbol repetition. Matching occurrences are corrected and the resulting sequence is accordingly fed to the GP toxicity model.

We defined two correction schemes, namely a deobfuscation filter or a neutralisation filter, depending on the kind of attack, obfuscation or polarity, respectively. In the former, the vocabulary consists of a list of toxic

![Figure 2. Disarming the adversarial attacks. The adversary comment is preprocessed in order to provide a suitable input to the toxicity model. (a) Obfuscation attack: the preprocessor is a deobfuscation filter fed with a list of toxic terms; if any obfuscated variant is found, the filter replaces them with their plain English version. (b) Polarity attack: the preprocessor is a neutralisation filter fed with a list of negation predicates; if any of these are found, in this case the filter removes them from the input text.](image-url)
terms that when obfuscated are not recognised by the model, thus lowering the toxicity of the comment whilst still conveying its aggressive tone; therefore here the filter corrects the input by replacing obfuscated occurrences with their corresponding plain English versions. In contrast, the neutralisation filter uses as vocabulary a list of negated predicates of such toxic terms; here the input is corrected by removing (i.e. pruning) plain or even obfuscated occurrences of these predicates, effectively switching the polarity of the comment towards a neutral attitude by expurgating references to (negated) toxic content. These two vocabularies are shown in Table 2.

3 Results

3.1 Experimental setup

In order to apply the toxicity model and the deobfuscation filter we used the Perspective Web API [1] and the TextPatrol Web API [2] respectively. For this purpose we developed a command-line tool that takes as input a dataset of comments that are processed by invoking the APIs in order to produce an output text file including, for each comment, toxicity scores and execution times of the GP model alone and the combined TP+GP method. The command-line tool was written using the Go programming language, it is open source and publicly available in the following Gitlab repository [https://gitlab.com/textpatrol/gp-tp-experiment], which also hosts the obfuscation and polarity attacks datasets, the set of result files and instructions on how to build the tool and re-run the experiments.

3.2 Obfuscation attack

Let us examine first the results of toxicity scores reported in Fig. The figure comprises three plots corresponding to the three topics: usel*, brex* and clic*. Each plot in turn combines the scores obtained in both datasets, obfuscation-50 and obfuscation-99, within two reflected panels (upper and lower half, respectively; notice that the y-axis scale ranges from 0 to 1 in both directions). Three sets of values are depicted in these plots: toxicity of the original comments (red squares) and average toxicities of the 1000 variants of each comment, as obtained by the GP model (green bars) and by the proposed TP+GP method (amber bars, whiskers being standard deviations).
Figure 3. Toxicity scores for the obfuscation attack. In each plot the upper half panel shows average toxicities obtained for comments obfuscated with a 50% corruption rate (\texttt{obfuscation-50} dataset) whereas the reflected lower half panel reports toxicities with a 99% corruption rate (\texttt{obfuscation-99} dataset). Bars indicate the average was taken over 1000 variants in their respective comment category (whiskers being standard deviations) whilst squares indicate the toxicity values of a single observation, i.e. those of the original comments. (a) US Election. (b) Brexit. (c) Climate change.
We initially note that the toxicity of the original deobfuscated comments (red squares) is nearly 1.0 for every comment, except usel09 and usel10 scoring toxicities closer to 0.9. These two categories contain toxic terms related to superlatives or adjective derivations (racist, supremacist, sexist, stupidity) whereas the remainder comments incorporate either insults or obscenity (idiot, stupid, asshole, moron, screw you) suggesting that the GP toxic model stresses profanity terms.

Next, we identify a common pattern on the results obtained by the GP model alone, namely its vulnerability to the obfuscation attack. This is no matter of surprise as this weakness in the model was previously suggested in the preliminary study of [6] using a few examples of the attack. Here it can be seen that in average, the GP model (green bars) decreases toxicities to levels closer to or lower than 0.5, which can be considered a moderate aggressiveness cut–off to decide if a comment is safe (18 out of 24 cases are below this cut–off in obfuscation–50 and 22/24 in obfuscation–99). A more stringent cut–off of 0.75 would have resulted in all but one of the toxic obfuscated categories been scored as non–toxic. Altogether, these findings substantiate on a large–scale basis that the adversarial obfuscation attack is effectively able to deceive the GP model.

On the other hand, we observe that in average the TP+GP method (amber bars) manages to restore the toxic scores close to their original values. On the obfuscation–50 dataset the differences between the TP+GP scores and the original toxicity differ in less than 0.03 in all but three cases: brex01 (0.11), brex07 (0.11) and clic07 (0.15). Similarly, on the more deceitful obfuscation–99 dataset, where essentially the entire sequence of toxic terms occurrences are obfuscated, the reduction in the scores is again less than 0.03 in all but four cases: usel09 (0.05), brex01 (0.21), brex07 (0.22) and clic07 (0.26). Notice that even with a stringent 0.7 cut–off, the proposed method in average will correctly identify all the cases as toxic. Besides, the evidence indicate that also in all cases, the average drop of toxicity incurred by the GP model is significantly larger than that the average drop obtained by the TP+GP method ($p < 0.001$).

In order to further compare the effectiveness of the proposed method in disarming the obfuscation attack, we illustrate in Fig. 4 radial frequency histograms of the proportion of obfuscated instances correctly scored with at least the original toxicity, from the total 1000 comments (red area) in each category. It can be seen how on the obfuscation–50 dataset the TP+GP method (amber area) is able to correctly score a large proportion of comments
Figure 4. Effectiveness in disarming the obfuscation attack. These radial histograms show the proportion of obfuscated comments correctly scored with at least the same toxicity of the original comment, out of 1000 variants (red area), as obtained by GP model (green sector) and TP+GP method (amber sector). (a) On the obfuscation-50 dataset. (b) On the obfuscation-99 dataset.

in all topics. This pattern is replicated in the obfuscation-99 dataset, although proportions slightly decline due to the higher corruption level, which yields more acid attacks.

Additionally, it is interesting to note that on the obfuscation-50 dataset the GP model (left panel, green area) is able to recognise and score correctly a small proportion of the variants, particularly in the use106, brex05 and clic01 categories. On further examination we found that the toxicity of the two former comes from the use of the insulting vocable “moron” (see Appendix A. Original comments). Now notice that the homglyphs defined for this particular combination of letters were simply their upper- and lowercase versions (see Fig[1]); thus, given a low corruption rate it is very likely that no segmentators are inserted and therefore some of the generated variants would consist of the same text spelled with intermixed case (e.g. “mORoN”,
“MoRON”, etc). In such instances, the attack can be easily disarmed by a simple case-converting operation which we assume the GP API is carrying out before feeding the comment to the model.

Regarding the recognition ability of the clic01 variants category by the GP Model, a closer inspection shows that its toxicity is originated from using the term “ass” in its offensive meaning (see Appendix A. Original comments); although in this case homoglyphs for these combination of letters include non-letter symbols (see Fig.1), we observe that because of the short length of the offensive term (3 character-long) along with the low corruption rate (50%) it is very likely that a number of generated variants are identical to the original or a mixed-case version as before. On the other hand, in the more acid obfuscation-99 attacks (right panel) harder obfuscated variants will be found more frequently, hence the dilution of the green bars.

Moreover, we observe that the longer the toxic sequence and the more toxic terms are included in the original comment, the stronger the deception effect of the obfuscation attack. Take for example use110 which contains 3 toxic terms, one of which is 12-letters long (see Appendix A. Original comments); in this case, the proportion of recognised variants by the GP model is tiny. Combine that with more frequent bogus segmentations due to higher corruption rates, resulting in the toxic recognition ability of the GP model to further weaken\(^1\), as it is noticeable in the radial histogram of the obfuscation-99 dataset. The TP+GP approach in contrast, is able to still recognise most of them despite the intensity of the attack.

Let us focus now on processing time. Fig. 5 shows average runtimes of the 1000 repetitions for GP model and TP+GP method in each category. The first observation here is that the effectiveness of the proposed method in disarming the obfuscation attack, comes at the expense of an increase in runtime. This is due of course to the additional preprocessing step that executes the deobfuscation filter. The trend in the plot is that the TP+GP nearly doubles the time taken by GP model alone.

The second observation is that the average runtimes for each comment category are quite similar in both obfuscation-50 and obfuscation-99, a reasonable behaviour considering that the computational complexity of the

\(^1\)Although it is well-worth noting that GP keeps refining its model by continuously learning from more examples, thus fortifying toxic scores appropriately. In fact, the toxicities recorded when this study began, soared up during the following months (see Appendix C. GP Model toxicity update). Here we report updated scores at the time of submission.
Figure 5. Average runtimes for the obfuscation attack experiment, GP model (green) vs. TP+GP method (amber). (a) obfuscation-50. (b) obfuscation-99.

deobfuscation method depends only on the length of the text [14]. Since the obfuscation attack may add extra characters (bogus segmentators or fake letter repetitions) only to the toxic terms within the comment, the total text length may vary but not drastically. Hence, the resemblance of average runtimes patterns exhibited in both datasets.

The last observation is related to the variability of the results, indicated by the whiskers in the plots. At first glance these deviations may seem large; nonetheless recall that the experimental pipeline was built using two web-service APIs, one for GP and another one for TP, respectively. In such setting the execution of the experiments need to take into account network latencies and server availability, which are less certain than having a dedicated host.
running the scripts\textsuperscript{2}. Besides, since in our experimental setup the TP+GP method is actually invoking two independent web services, it is more likely to suffer further delays that affect its runtime variability.

### 3.3 Polarity attack

The results of the polarity attack experiment are summarised in Fig. \ref{fig:polarity}, where toxicities for the original comments and the negated variants using both GP and TP+GP are shown; recall that the polarity dataset contains a single negated variation per comment, hence the figure reports a single run per category (\{\texttt{use101'},\ldots,\texttt{clic07'}\}). These scores indicate that this attack also manages to deceive the GP model, because the negated versions were assigned high toxicities (although lower than their respective original comments). When using the TP+GP method on the contrary, the toxicities fall to levels mostly below 0.5, except for some comments that incidentally contain toxic content not initially identified as such at the beginning of our study, but incorporated into newer versions of the GP model (e.g. terms like “dishonest” in \texttt{use104} or “short-sighted” in \texttt{clic06}, that do not appear in the vocabulary lists of Fig. \ref{fig:vocab}).

The rationale of this behaviour is that the toxicity model is trained on the assumption that the comments would be coherently written, in a grammatical sense. Thus it simply checks whether toxicity features and patterns are identified within the comment and not if it is correctly constructed. To illustrate this point, let us analyse the effect of the neutralise filter on comment \texttt{use101'}: \textit{If they voted for Hilary they are NOT idiots}. Using GP, this comment scores a 86% toxicity score (very close to the score of the original aggressive comment), which when removing the negated offensive term (\textit{If they voted for Hilary they are}) plunges to a 25% toxicity level; in other words, GP would identify the trimmed sentence as healthy, despite its dubious grammar soundness.

The TP+GP approach takes advantage of the aforementioned assumption, by simply removing subsequences matching the list of negated predicates. One may argue however, that the trimmed comment is not comparable with the negated one, as the former may become a grammatically malformed sentence (due to the removed predicates). The reason we proceeded that

\footnote{In some cases the GP web service replied with a 400 Bad Request or 502 Bad Gateway error codes; these cases were excluded from the reported average runtimes (651 in \texttt{obfuscation-50} and 506 in \texttt{obfuscation-99} out of their corresponding 24,000 variants).}
Toxicity scores for the polarity attack, as obtained by GP model (green) vs. TP+GP method (amber) in the 24 negated comments of the polarity dataset. Toxicities of the original comments are also shown (red).

Figure 6.

way is we wanted to preserve as far as possible the authentic source content, allowing the filter to remove but not to alter or add new content ad libitum. An alternative approach would have been precisely to replace, instead of to remove, the matching negated predicates with non-negated synonyms (e.g. not awful → wonderful, not idiot → clever); as a result, correctly constructed toxic-trigger-free comments would have been obtained. Hence, their scores would also have dropped down to the healthiness region (in this example If they voted for Hilary they are clever, obtained a 24% toxicity, almost equalling the trimmed sentence).

As a closing remark, runtimes of this experiment resembled those of the obfuscation attack in the sense that TP+GP nearly doubled GP processing time (data not shown).

4 Conclusion

Machine learning models of language toxicity such as the GP engine, are effectively able to estimate aggressiveness in written comments. Nonetheless, these models can be deceived by adversarial manipulation of the text, misusing the linguistic assumptions on which they are trained. Firstly at the character-level, an adversary may take advantage of the versatility of writ-
ten text along with the robust reconstruction capacity of the human brain, to replace similar graphemes and/or to insert fake segmentations yielding toxic contents unrecognisable to the model, thus increasing its false negative rate. Secondly, at the sentence level, an adversary may deceive the model using grammatical operations as simple as negations, switching the polarity of the comment but keeping a structure similar to previously learned toxic examples, hence, increasing the model false positive rate.

These attacks can be alleviated by preprocessing the comment so as to restore its standard sequence and/or polarity representation before feeding the toxic model. In the first kind of attack, such an approach can be effective even if half the length of the toxic sequence is obfuscated; higher corruption rates decrease to some extent its effectiveness, but then also the deception attempt becomes futile since the obfuscation becomes illegible to human readers. In the second kind of attack, by removing the negated toxic predicates or replacing them with affirmative synonyms, the preprocessor is able to restore the correct toxicity; however we anticipate further research is required since negation is a complex grammatical category whose rules differ amply between lexical elements such as verbs compared to adjectives, nouns and other clauses.

Another avenue of future study may consider harder adversarial attacks such as transposition or tandem–character obfuscations, that despite losing some of the visual deception factor still pose alluring challenges (e.g., `fuck → fukc`, `SHIT → S|-|IT`). Besides, computational speedups in the execution of the filters are also of practical interest (see [10] or more recently, [11]), considering that language is an entity in constant evolution[5] whose toxic vocabulary evolves likewise. Moreover, it would be feasible to extend our adversarial toxic comments detection to other Latin–based languages by training their respective word- or character-level n-grams machine learning models [17] coupled with filters of obfuscation–prone customised vocabulary [14]. Lastly, exploring feature spaces derived from conversational networks [13] may improve the effectiveness of our methods.

As a final remark, we believe that ensuring aggression–free, respectful and opinionated online discussions would require a pipeline of text processors working on different levels of abstraction, ranging from sequence to linguistic to semantic to sentiment analysis; lower level processing would provide useful features to higher level analysis, which of course, may prove more difficult to handle. To illustrate this point, take for example figures of speech such as sarcasm, irony or satire that may use toxic content as a legitimate rhetorical
device. In any case, aggression is strongly related to arising causes of concern such as cyberbullying, harassment, post-truth opinion forming, pedophilia, terrorism and mind state manipulation towards violence [16, 3, 8]; hence, robust toxicity detection will have significant implications for casual, legal or political issues in online communities.

**Disclaimer:** Original comments and modified variants, as depicted in Tables 1 and 3 and in other sections of this paper, were collected or derived from their original source [1] for research purposes only; they do not represent the views or opinions of the authors.

**References**

[1] Perspective API. https://www.perspectiveapi.com. Accessed: 2017-09-24.

[2] TextPatrol API. https://api.textpatrol.tk. Accessed: 2017-09-24.

[3] Pete Burnap and Matthew L Williams. Us and them: identifying cyber hate on twitter across multiple protected characteristics. *EPJ Data Science*, 5(1), 2016.

[4] Robert Dale. NLP in a post-truth world. *Natural Language Engineering*, 23(2):319–324, 2017.

[5] Hal Hodson. Our future language? English rulz OK, txtspk and mind-reading. *New Scientist*, 2016.

[6] Hossein Hosseini, Sreeram Kannan, Baosen Zhang, and Radha Poovendran. Deceiving Google’s Perspective API Built for Detecting Toxic Comments. *arXiv preprint arXiv:1702.08138*, February 2017.

[7] Homa Hosseinzamiri. Survey of computational methods in cyberbullying research. In *Proceedings of the First International Workshop on Computational Methods for CyberSafety*, New York, NY, USA, 2016. ACM.

[8] Homa Hosseinzamiri, Sabrina Arredondo Mattson, Rahat Ibn Rafiq, Richard Han, Qin Lv, and Shivakant Mishra. Analyzing labeled cyberbullying incidents on the instagram social network. In Tie-Yan Liu,
Appendixes

Appendix A. Original comments

Table 3 shows the text sequences of the original toxic comments extracted from the GP Website [1] with their toxicity scores at the beginning of this study.
Table 3. The list of original toxic comments extracted from [1].

| ID    | Comment sequence                                                                                                                                                                                                 | Toxicity | Length |
|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------|--------|
| use101| If they voted for Hilary they are idiots                                                                                                                                                                | 0.90     | 40     |
| use102| It was awful. People are stupid.                                                                                                                                                                              | 0.90     | 32     |
| use103| I respect it but they are stupid                                                                                                                                                                             | 0.89     | 32     |
| use104| Liberals are idiots. Why in the hell would u vote for a person like Hillary Clinton. She believes in stealing, she’s dishonest and believes it’s okay for an abortion to be done at 9 months. She’s awful. | 0.87     | 202    |
| use105| Donald trump is the worst person to be president. This election really showed how stupid the U.S. is.                                                                                                       | 0.85     | 101    |
| use106| Anyone who voted for Trump is a moron.                                                                                                                                                                         | 0.81     | 38     |
| use107| Screw you trump supporters                                                                                                                                                                                   | 0.79     | 26     |
| use108| It was a terrible. Both sides suck, but Trump REALLY is scary                                                                                                                                                | 0.72     | 61     |
| use109| It is a shame that Donald Trump was elected. You can never underestimate the stupidity of middle America.                                                                                                 | 0.67     | 105    |
| use110| Good job voting for a racist, sexist guy who’s putting white supremacists in the white house. Have fun.                                                                                                      | 0.67     | 103    |
| brex01| Anti democratic assholes                                                                                                                                                                                    | 0.95     | 24     |
| brex02| Ignorant and stupid                                                                                                                                                                                          | 0.93     | 19     |
| brex03| They are stupid and ignorant with no class                                                                                                                                                                  | 0.91     | 42     |
| brex04| It’s stupid and wrong                                                                                                                                                                                        | 0.89     | 21     |
| brex05| You morons                                                                                                                                                                                                  | 0.92     | 10     |
| brex06| Idiots. backward thinking people. nationalists. not accepting facts. susceptible to lies.                                                                                                                  | 0.80     | 89     |
| brex07| It’s rubbish                                                                                                                                                                                                  | 0.69     | 12     |
| clic01| They have their heads up their ass.                                                                                                                                                                         | 0.93     | 35     |
| clic02| How can you be so stupid?                                                                                                                                                                                   | 0.91     | 25     |
| clic03| They are liberal idiots who are uneducated.                                                                                                                                                                   | 0.90     | 43     |
| clic04| They’re stupid, it’s getting warmer, we should enjoy it while it lasts.                                                                                                                                       | 0.86     | 71     |
| clic05| Climate change is happening and it’s not changing in our favor. If you think differently you’re an idiot.                                                                                                | 0.84     | 105    |
| clic06| I think those people are stupid and short-sighted                                                                                                                                                             | 0.84     | 49     |
| clic07| They’re allowed to do that. But if they act like assholes about, I will block them.                                                                                                                     | 0.78     | 83     |
Appendix B. Datasets and experiment source code

The obfuscation-50, obfuscation-99, and polarity datasets as well as the Go language source code used in our experiments are publicly available at the following Gitlab repository: https://gitlab.com/textpatrol/gp-tp-experiment.

Appendix C. GP Model toxicity update

Fig. 7 shows a comparative plot of the toxicity scores given by the GP model at the beginning of the study (February, 2017) versus at the time of submission (September, 2017). For the original comments it can be seen that scores have raised, suggesting that the toxicity model has been refined during this period, probably by learning from more examples and newer toxic contents. On the contrary, the negated comments show a trend of decreasing toxicities; we believe the model has been refined to account for additional linguistic features related to the grammatical structures of negation, although still it is being deceived by this attack to obtain scores higher than 50%.

Figure 7. GP toxicity scores variation during the lasts months (Feb-Sep/2017).