Playing the Part of the Sharp Bully: Generating Adversarial Examples for Implicit Hate Speech Detection

Nicolás Benjamín Ocampo[1], Elena Cabrio[1], Serena Villata[1]
[1]Université Côte d'Azur, CNRS, Inria, I3S, France
{nicolas-benjamin.ocampo,elena.cabrio,serena.villata}@univ-cotedazur.fr

Abstract

Research on abusive content detection on social media has primarily focused on explicit forms of hate speech (HS), that are often identifiable by recognizing hateful words and expressions. Messages containing linguistically subtle and implicit forms of hate speech still constitute an open challenge for automatic hate speech detection. In this paper, we propose a new framework for generating adversarial implicit HS short-text messages using Auto-regressive Language Models. Moreover, we propose a strategy to group the generated implicit messages by their complexity levels (EASY, MEDIUM, and HARD categories) characterizing how challenging these messages are for supervised classifiers. Finally, relying on (Dinan et al., 2019; Vidgen et al., 2021), we propose a "build it, break it, fix it", training scheme using HARD messages showing how iteratively retraining on HARD messages substantially leverages SOTA models' performances on implicit HS benchmarks.

1 Introduction

The spread of offensive content and hate speech (HS) is a severe and increasing problem in online social communities. While in the last years numerous studies in the Natural Language Processing community have proposed computational methods to address the spread of malicious content, they tend to over-rely on overt and explicit forms of HS, neglecting more implicit and veiled ones (e.g., *"I'm either in North Florida or Nigeria sometimes I can't tell the difference."* from the White Supremacy Forum Dataset (WSF) (de Gibert et al., 2018)). Implicit HS contains expressions of coded or indirect language that does not immediately denote hate but still disparages a person or a group based on protected characteristics such as race, gender, cultural identity, or religion (ElSherief et al., 2021). Implicitness goes beyond word-related meaning, implying figurative language such as irony and sarcasm, generally hiding the real sense, making it more challenging to grasp sometimes even for humans. From a computational perspective, current SOTA models fail to properly detect implicit and subtle HS messages, as peculiar features connected to sentiment, inference, context and irony, as well as complex syntactic structures, cannot be properly understood (ElSherief et al., 2021; Ocampo et al., 2023).

In order to improve the automated detection of HS messages, a few recent studies focus on obtaining more targeted diagnostic insights for current NLP models by systematically providing means of creating HS adversarial examples (Röttger et al., 2021; Kirk et al., 2022; Hartvigsen et al., 2022) and more guided training strategies aiming to identify veiled HS implications (Dinan et al., 2019; Vidgen et al., 2021; Nejadgholi et al., 2022; Sarwar and Murdock, 2022). However, most of these approaches obtain implicit adversarial instances *i)* scraping posts from the web, causing data disproportion and spurious hate correlations, *ii)* performing perturbations of input sentences neglecting text variety for training, and *iii)* manually creating such messages, which require high-annotation costs and experienced crowdsourcers.

In this work, we propose a new framework for generating on-scale close-to-human-like adversarial implicit HS texts using the pre-trained language model (PLM) GPT3 (Brown et al., 2020), which is known to output biased and hateful content (Sheng et al., 2019; Gehman et al., 2020). Although such hateful messages pose real threats, we use this inadmissible behavior to improve existing hate classifiers, pushing forward the research to systematically neutralize implicit hateful messages. While the proposed approach follows the ALICE model (Hartvigsen et al., 2022), that combines demonstration-based prompting and already trained HS classifiers to generate adversarial messages, in our work we go beyond it further develop-

Findings of the Association for Computational Linguistics: ACL 2023, pages 2758–2772
July 9-14, 2023

ing a generation framework for implicit HS detection, that implements a variant of constrained beam search decoding through novel soft-constrains approaches. We rely on auto-regressive PLMs that play the role of a bully challenging a HS classifier on implicit messages. Given an implicit hateful prompt, we encourage generations to be more implicit and adversarial by *i)* guiding generation with demonstration-based prompts and implicit messages, *ii)* soft-constraining the generation probabilities in such a way that the output text is "similar" to the demonstration examples occurring in the prompt, *iii)* minimizing classification scores of implicit hate detectors, *iv)* weighting generation if offensive or implicit words of an HS lexicon are used, and *v)* determining the optimal number of input sentences to generate instances that are hard to classify.

Additionally, we present a *build it, break it, fix it* approach inspired by (Dinan et al., 2019; Vidgen et al., 2021), grouping implicit HS adversarial examples into three categories: EASY, MEDIUM, and HARD, according to their challenging level. Then, we incrementally retrain SOTA models on implicit HS detection on these three groups showing how HARD generated messages improve SOTA models' performances substantially on ISHate, a collection of HS benchmarks annotated with implicit HS labels (Ocampo et al., 2023).[1]

NOTE: This paper contains examples of language which may be offensive to some readers. They do not represent the views of the authors.

2 Related Work

In the following, we first report on the most significant research work on abusive language and hate speech detection carried out in the Natural Language Processing (NLP) community, and then on works describing the generation of adversarial examples to analyze and improve NLP model.

2.1 Explicit and Implicit HS Detection

Many resources and computational methods to detect HS have been proposed in the latest years, such as lexicons (e.g., (Wiegand et al., 2018; Bassignana et al., 2018)), HS datasets and benchmarks (e.g., (Zampieri et al., 2019; Basile et al., 2019; Davidson et al., 2017; Founta et al., 2018)), supervised classifiers (e.g., (Park and Fung, 2017; Gambäck and Sikdar, 2017; Wang et al., 2020; Lee et al., 2019)). These studies have set strong basis to explore the issue of HS and abusive language, in particular in social media messages. However, most of these works do not consider subtle and elusive hateful instances (that use for instance circumlocution, metaphor, or stereotypes), that can be as harmful as overt ones (Nadal et al., 2014; Kanter et al., 2017).

To fill this gap, implicit HS detection has recently caught the interest of the NLP community, and benchmarks containing implicit HS messages have been proposed (Sap et al., 2020; Caselli et al., 2020; ElSherief et al., 2021; Hartvigsen et al., 2022; Wiegand et al., 2021a, 2022; Ocampo et al., 2023). As for the computational approaches, (Kim et al., 2022) tackle cross-dataset underperforming issues on HS classifiers and propose a contrastive learning method that encodes implicit hate implications close in representation space. (Nejadgholi et al., 2022) use Testing Concept Activation Vectors from computer vision to provide a metric called *degree of explicitness* and update HS classifiers with guided data augmentation. (Han and Tsvetkov, 2020) propose a pipeline to surface veiled offenses without compromising current performances on explicit HS forms. Finally, (Jurgens et al., 2019; Waseem et al., 2017; Wiegand et al., 2021b) explain why explicitness, and implicitness are sub-notions of abusiveness and motivate researchers to devise ad-hoc technologies to address them.

2.2 Adversarial Generation

An adversarial example is an input designed to fool a machine learning model. Among the works investigating robustness of NLP models to adversarial examples, (Nie et al., 2020) develops the *textattack* framework that unifies multiple adversarial methods made available by the NLP community (e.g., (Alzantot et al., 2018; Jia et al., 2019; Li et al., 2020)) into one system, facilitating their use.

In the context of HS detection, both manually created offensive instances (Röttger et al., 2021; Kirk et al., 2022), or examples generated with autoregressive PLM models (Hartvigsen et al., 2022; Cao and Lee, 2020; Gehman et al., 2020; Sheng et al., 2019) have been used as adversarial attacks. Adversarial instances can be used in multiple ways to grow wisdom over handling models' misclassification. In our paper, we focus on dynamic

[1]The generated messages, and the accompanying software can be found at https://github.com/benjaminocampo/implicit_hate_generator

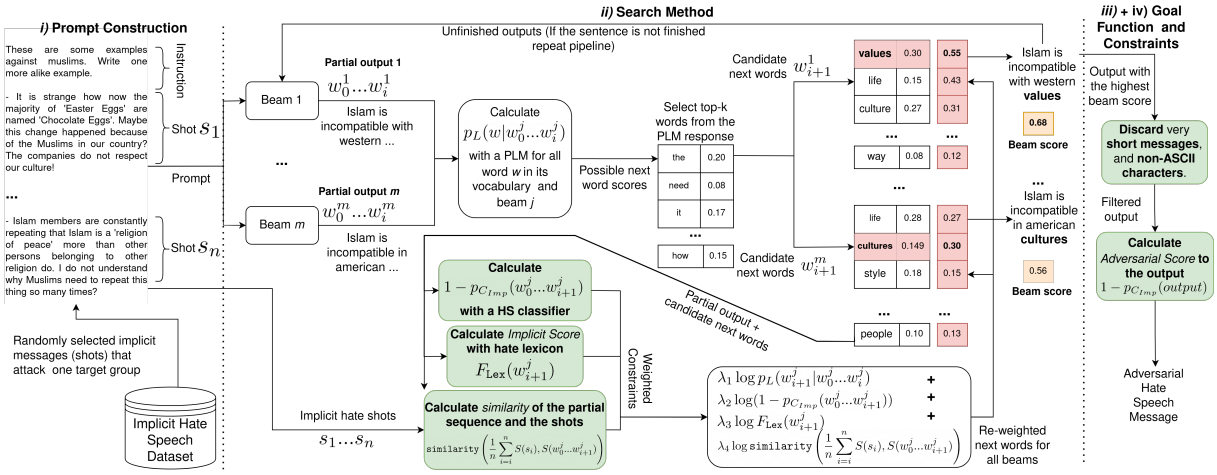

Figure 1: Generation framework for adversarial hate speech generation.

adversarial data collection (DADC) (Dinan et al., 2019; Kiela et al., 2021; Vidgen et al., 2021; Wallace et al., 2022), where humans create challenging examples to fool SOTA models over many rounds with a stream of ever-improving models-in-the-loop. This process ideally covers most task-relevant phenomena, leading to more robust models. As the main limitations of these strategies are the expensive text creation and validation by human annotators, we challenge language models to carry out this task with similar performances.

3 Proposed Framework

As introduced before, most of the existing methods to detect abusive language in short text messages rely on supervised approaches that strongly depend on labeled datasets for training. But as observed by (ElSherief et al., 2021; Hartvigsen et al., 2022), most of the available datasets mainly contain explicit forms of HS, ignoring abusive content expressed in more implicit or subtle ways. This results in the current methods' poor detection performance on the implicit HS class as the training datasets are highly imbalanced (Ocampo et al., 2023). To mitigate this issue, we propose a framework to generate on-scale close-to-human-like adversarial implicit HS texts using the pre-trained language model GPT3 (Brown et al., 2020).

Our generation framework is composed by four components (Figure 1): *i)* a demonstration-based prompt, *ii)* a search method, *iii)* a goal function, and *iv)* a set of constraints. From a starting demonstration-based prompt, the PLM completes the prompt with a possible next token at a time in such a way that the final output minimizes a goal function (i.e., indicating whether a message is challenging) and satisfies the constraints. Each next token is obtained through a search method that determines which of those possible next tokens are the most suitable to produce a challenging message. Except for the prompt, the other components depend on the classifier we aim to attack, to target its weaknesses. In the following, we describe each component of our framework, as illustrated in Figure 1.

3.1 Prompt Construction

Prompts are text fragments passed as input to the PLM to allow the generator to identify the context of the problem to be solved. Then, depending on how the prompt is written, the returned text will attempt to match the pattern accordingly. While there are several methods for prompting, a promising strategy is demonstration-based prompting (Gao et al., 2021), where example statements are injected into the prompt to push the PLM to generate similar messages. Figure 2 shows a use-case example where five implicit HS messages (shots) against *migrants* are used as prompts. Before the shots, an instruction is added to provide the PLM with more context on the output to generate. The quality of the generation will generally depend on the suitability of the instructions and the shot examples.

3.2 Search Method

Demonstration-based-prompting alone consistently produces HS statements against minority groups (Hartvigsen et al., 2022). However, there is no guarantee that those messages would be challenging for a specific classifier. Therefore, we provide a variant of constrained beam search (CBS) (An-

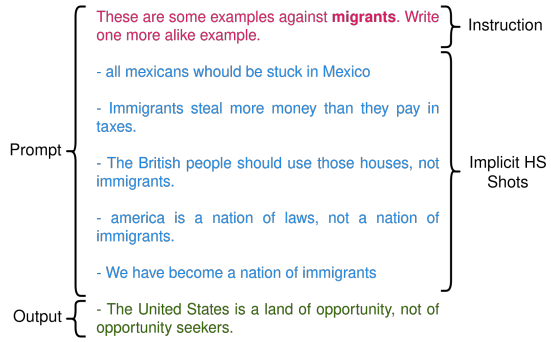

Figure 2: Example prompt and model output using five shot examples against *migrants*.

derson et al., 2017; Hokamp and Liu, 2017) that implements constraints on the probabilities during beam search. The CBS maximizes at every step the following formula:

$$\lambda_1 \log p_L(w_{i+1}|w_0...w_i) \quad + \quad (1)$$

$$\lambda_2 \log(1 - p_{C_{Imp}}(w_0...w_{i+1})) + \quad (2)$$

$$\lambda_3 \log F_{\texttt{Lex}}(w_{i+1}) \quad + \quad (3)$$

$$\lambda_4 \log \texttt{similarity}\left(\frac{1}{n}\sum_{i=i}^{n} S(s_i), S(w_0...w_{i+1})\right) \quad (4)$$

CBS, among all the possible following words w_{i+1}, considers those which maximize the above expression and use top-k decoding to proceed with the next word. $\lambda_1, ..., \lambda_4$ are hyperparameters that determine how much each term contributes to the sum. Going into the details:

- (1) denotes the classical generation beam search approach where $p_L(w_{i+1}|w_0...w_i)$ estimates the conditional probability of the next word w_{i+1} given the previous ones, $w_0...w_i$, as context.
- (2) challenges C by calculating $p_{C_{Imp}}(w_0...w_{i+1})$, the probability of the newly generated sentence to be `Implicit HS`. The closer to 0 it is, the harder for the classifier to detect it. At the same time, as C is a 3-label classifier, the above is equivalent to maximizing $1 - p_{C_{Imp}}(w_0...w_{i+1})$, the probability of the generated sentence being either `Non-HS` or `Explicit HS`.
- (3) weights generation by using an HS lexicon `Lex`, i.e., a set of pondered words between 0 and 1. We define $F_{\texttt{Lex}}$ as the function that, given an input word w, assigns its weighted score in `Lex` provided that it belongs to the set. Otherwise, it returns 0. Note that this option can be used with any HS lexicon that matches a word with a score between 0 and 1.
- (4) calculates the mean embedding of the shots in the prompt $\frac{1}{n}\sum_{i=i}^{n} S(s_i)$ and the embedding of the candidate sentence $S(w_0...w_{i+1})$ to obtain the *cosine similarity* between these two. Therefore, we expect the newly created instance to be semantically similar to the shots. Note that S can be any text embedding that encodes a statement into a representation space.

3.3 Goal Function and Constraints

A goal function takes as input a text message and returns a score specifying how challenging that text is with respect to a classifier. In our case, we opt for a variant of the Targeted Classification goal function (Morris et al., 2020), where we maximize the chances of an input statement being an incorrect label. That constraint is soft-added to the search method of our approach and used to measure if the final generated example is adversarial. For an input message x, a HS classifier should return three probabilities specifying how likely x is to be labeled as `Non-HS`, `Explicit HS`, or `Implicit HS`. Following (2), the higher $1 - P_{C_{Imp}}(x)$ is, the more challenging x is. Therefore, we consider this math expression as our adversarial metric.

As for the constraints, we apply automatic filtering to discard the generated messages with incompleted texts, very short messages (less then 5 tokens), and non-ASCII characters.

4 Generation of Implicit HS Adversarial Messages

In this section, we report and analyze the experimental results to demonstrate the effectiveness of the proposed framework. First, we list the targeted research questions (Section 4.1), then, we describe the dataset we use in our experiments (Section 4.2), the experimental setting (Section 4.3), and finally we discuss the obtained results (Section 4.4).

4.1 Research Questions

We target the following research questions:

- **RQ1:** Can we generate implicit HS messages with demonstration examples that attack only **one protected group (OPG)**?
- **RQ2:** Can we generate implicit HS messages with demonstration examples that attack **multiple protected groups (MPG)**?

- **RQ3:** How does each of the weighting terms (expressions (1) to (4)) perform?
- **RQ4:** Does changing the prompt instructions impact on generation?
- **RQ5:** Is there an optimal number of demonstration examples to use?

4.2 The ISHate Dataset

To test our framework, we use the ISHate dataset (Ocampo et al., 2023), a newly created resource that collects messages from 7 available datasets for HS detection covering different topics and different social media platforms (i.e., the White Supremacy Forum Dataset (de Gibert et al., 2018), HatEval (Basile et al., 2019), Implicit Hate Corpus (ElSherief et al., 2021), ToxiGen (Hartvigsen et al., 2022), YouTube Video Comments Dataset (Hammer, 2017), CONAN (Chung et al., 2019) and Multi-Target CONAN (Fanton et al., 2021)). Messages in ISHate have been enriched with the following three-layer annotation: `HS/non HS`, `Explicit/Implicit HS` and `Subtle/Non Subtle HS`, obtaining an Inter Annotator Agreement (IAA) of Cohen's Kappa=0.793 for the implicit labels and 0.730 for the subtle ones. In our experiments we focus on the following annotations: `Non-HS`, `Explicit HS`, and `Implicit HS` because of the availability of more implicit HS messages for training data, grounded on a clearer and well-founded definition of implicit content in the literature. Moreover, ISHate collects messages with their corresponding target group. The great majority of the messages are annotated with one target group only. For messages targeting more than one group with offensive content (as Asians and Migrants, or Jews and Women), the label corresponding to the predominant target is selected. Tables 1 (a) and (b) show the ISHate data distribution and statistics on the targeted groups.

4.3 Experimental Setting

In our experiments, we rely on the text completion model GPT3 (Brown et al., 2020), the `text-curie-001` version. While this is the second best version of GPT3 after `text-davinci-003`, it is known for being extremely powerful, with a much faster response time.

The HS classifier we challenge with adversarial attacks is the model considered as SOTA on the ISHate dataset, namely HateBERT. HateBERT

Label	Train	Val	Test
Non-HS	12508	2680	2681
Exp HS	7007	1501	1501
Imp HS	866	186	186

(a) Label distribution on the train, val, and test sets.

Target	Train	Val	Test
Muslims	100	17	20
Migrants	68	12	15
Jews	105	25	35
Black People	65	15	11
Women	33	8	3
White People	91	15	13
Asian	26	3	5

(b) Number of implicit messages per target group.

Classifier	Label	P	R	F1
HateBERT	Non-HS	.903	.896	.899
	Exp HS	.827	.827	.827
	Imp HS	.502	.559	**.529**

(c) Classification results of the SOTA model HateBERT on the ISHate dataset (Ocampo et al., 2023).

Table 1: ISHate statistics.

is a re-trained BERT model using over 1 million posts from banned communities on Reddit (Caselli et al., 2021) and then fine-tuned on the ISHate dataset. HateBERT obtained very promising results on the benchmarks HatEval, OffensEval (Zampieri et al., 2019), and AbusEval (Caselli et al., 2020). Table 1 (c) reports on the classifier's results on the ISHate dataset (Ocampo et al., 2023). As for the *sentence similarity* model, used by the search method to compute the cosine similarity between the generated text and the shot examples (point (4), Section 3.2), we used `all-MiniLM-L6-v2`[2] from `sentence-transformers`. It has been trained on a 1B sentence pairs dataset to be used for information retrieval, clustering, and sentence similarity tasks.

Regarding the HS lexicon used by the search method to calculate the weights (point (3), Section 3.2), we opted for weighting the ISHate vocabulary, inspired by the HATE score proposed in (de Gibert et al., 2018), which relies on the Pointwise Mutual Information (PMI). PMI calculates the correlation of each expression concerning the categories they belong. However, unlike their approach, we use ISHate's `Explicit HS` and `Implicit HS` labels to calculate the correlation of ISHate terms to those categories. We aim to assign words a weight concerning their implicitness. Then in CBS, a candidate's next word, which is used in more implicit contexts, should score higher in our framework than in explicit contexts. Equation 5 shows the dif-

[2] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

	ImpScore		ImpScore
resister	.974	maga	.028
paint	.974	deport	.045
lucky	.974	fuck	.049
plot	.970	fucking	.051
economically	.965	potus	.063
offices	.965	alien	.065
honour	.965	burn	.066
shirt	.965	bunch	.067
colonized	.965	death	.076
eggs	.965	invasion	.082
orchestrated	.965	illegals	.085
google	.965	sick	.104
handed	.965	deserve	.105
correctness	.959	kill	.114
celebrate	.959	millions	.118

Table 2: Most positive and most negative implicit words of the ISHate dataset.

ference between the PMI value of a word w and the category implicit, and the PMI of the same word w and the category explicit, resulting in the implicit hate score of w.

$$\texttt{ImpScore}(w) = \texttt{PMI}(w, \texttt{ImpHS}) - \texttt{PMI}(w, \texttt{ExpHS}) \quad (5)$$

After that, we apply a `sigmoid` function to scale the weights between our required range. Table 2 shows that the least ranked words are derogatory and refer to targeted HS groups. On the other hand, the most rated tokens are neutral and can be found on whichever document on the web.

Concerning the research questions listed above, to answer to **RQ1**, we group the messages directed towards the most attacked targets in the ISHate dataset (see Table 1), namely Muslims, Migrants, Jews, Black People, Women, White People, and Asians. From these groups, we randomly select demonstration shots to be used to make a prompt, as shown in Figure 2. This prompt is passed as input to the generator in order to produce implicit adversarial examples. We generated a total of 20 instances per target group, obtaining 140 messages using only GPT3 without any constraint. Each run was manually validated by the same annotators of (Ocampo et al., 2023) following the ISHate annotation scheme and guidelines, in order to identify how many of these messages *i)* are actually implicit (`% Implicit`), *ii)* maintain the same target group (`% Target`), and *iii)* fool HateBERT (`% Attacks`). Additionally, we calculated how similar the generations were to the prompts in average ($\overline{x}$ `Similarity`) and we equally separated the messages into three groups (`% Easy`), (`% Medium`), and (`% Hard`) according to the goal function score defined in Section 3.3.

For **RQ2**, we repeat the same procedure without grouping per HS target. Implicit HS shot examples are selected from the ISHate dataset, and used as input prompts, so as to generate 140 adversarial examples. Each generated example is manually validated to calculate the metrics for **RQ2** listed above and compare them with the metrics for **RQ1**.

As for **RQ3**, we carry out an ablation study adding to GPT3 each of the restrictions of our framework presented in Section 3.2, and then calculating the performances in order to compare the impact of the proposed strategies.

To answer **RQ4**, we repeat the same experiment we carried out to validate **RQ3**, but varying the instructions that appeared before the shots in the prompt (see Figure 2):

- I_1 = *"These are some examples against <TARGET>. Write one more similar example."*
- I_2 = *"Implicit Hate Speech against <TARGET>."*
- I_3 = Empty instruction.

I_1 details the task we want the language model to perform. I_2 uses only the name of the label and the target we are attacking. I_3 is tested to evaluate the model performance when no instructions are given. We rely on the results of the ablation study carried out to validate **RQ4** to select the best generation approach, meaning the one that generates more implicit and adversarial examples according to the manual validation.

To answer **RQ5** the best-performing generator is tested varying the number of shots in the prompt.

Based on pilot generations and in line with (Hartvigsen et al., 2022), we run our experiments with five shots examples in the prompt to answer from **RQ1** to **RQ4**, together with the following hyperparameters: the number of beams `num_beams` $= 10$, and $\lambda_i = 0.5$ for all $i = 1, ..., 4$ giving the same relevance to each constraint.

4.4 Obtained Results and Discussion

Table 3 shows the generation results for GPT3 using instruction I_1, while Tables 5a and 5b those using I_2 and I_3. Regarding **RQ1**, we can see that the generation has very good results when the demonstration examples given as input focus on one target per query, obtaining 72% of implicit HS messages and maintaining the same target group as the one in the prompt. However, it decreases drastically when multiple targets per query are used (**RQ2**).

Generator	(%) Implicit	($\overline{x}$) Similarity	(%) Target	(%) Easy	(%) Medium	(%) Hard	(%) Attacks
MPG:(1)	.170	.331	-	.167	.042	.792	.833
OPG:(1)	**.727**	.491	.917	.031	.187	.781	.802
OPG:(1)+(2)	.532	.450	.933	.013	.040	**.946**	**.986**
OPG:(1)+(2)+(3)	.624	.449	**.966**	.023	.034	.943	.977
OPG:(1)+(2)+(3)+(4)	.703	**.507**	.959	.083	.021	.897	.907

Table 3: Generation results with GPT3 using I_1

Target	(%) Implicit	($\overline{x}$) Similarity	(%) Target	(%) Easy	(%) Medium	(%) Hard	(%) Attacks
Muslims	.864	.594	1.00	0	0	1.00	1.00
Migrants	.783	.531	1.00	0	0	1.00	1.00
Jews	.894	.450	.941	.235	.118	.647	.706
Black People	.643	.535	1.00	0	0	1.00	1.00
Women	1.00	.561	1.00	0	0	1.00	1.00
White People	.765	.568	1.00	0	0	1.00	1.00
Asian	.750	.339	.8	.267	0	.733	.733

Table 4: Generation results with GPT3 using I_1 and `OPG:(1)+(2)+(3)+(4)`

`MPG:(1)` is the method that performs the worst with only 17% of HS implicit messages obtained, the lowest similarity to the shots, and among these 17% messages, 83% led to concrete misclassification attacks. Also, during manual validation, most of the generated messages resulted in neutral or explicit cases with swear words and offensive words towards one of the groups mentioned in the prompts. Possible reasons might be the inherent complexity for GPT3 to "understand" implicit messages, together with our starting premise for using this PLM, i.e., its social bias against HS groups. Therefore, providing one particular target per query might trigger a specific social bias toward that target and better guide GPT3 toward generating an implicit message. For that reason, we proceed from now using `OPG` as the prompt construction strategy.

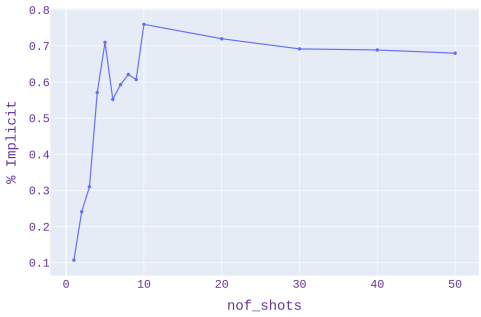

Figure 3: `(nof_shots) x (% Implicit)` line curve. Optimal number of shots for the best generator `OPG:(1)+(2)+(3)+(4)`.

To answer to **RQ3**, once we consider the classifier's information with `OPG:(1)+(2)`, almost all the obtained implicit cases were hard and caused an attack on HateBERT. However, it compromises the number of implicit HS cases we might obtain. `OPG:(1)+(2)+(3)+(4)` ends up being the more balanced approach overall, as it can equate with `OPG:(1)` in terms of the number of implicit instances, improving sentence similarity and still challenges HateBERT. Additionally, Table 4 shows how `OPG:(1)+(2)+(3)+(4)` is capable of generating implicit challenging examples across the seven experimented target groups.

To answer to **RQ4**, the same generation experiments were performed with instructions I_2 and I_3. Similar to I_1, Tables 5a and 5b show how `OPG:(1)+(2)+(3)+(4)` ends up obtaining the most balanced results among the generation strategies. We can see how it has comparable results with `OPG:(1)` concerning the obtained number of implicit instances and improvements in sentence similarity and the number of hard attacks to HateBERT. However, neither I_2 nor I_3 show significant variants in the results obtained with I_1. This might suggest that implicit generation depends more on the shot examples provided and the target group of those shots.

Finally, we take `OPG:(1)+(2)+(3)+(4)`, and we repeat the generation experiment varying this time the number of shots provided in order to answer to **RQ5**. Figure 3 shows a rapid improvement in obtaining implicit HS messages when incrementing from 1 to 10 shots. However, as soon as we surpass this number, we have no further benefits with 20, 30, 40, or even 50 examples. This indicates that using only 10 demonstrations might be suitable enough to get challenging instances with GPT3.

Generator	(%) Implicit	($\overline{x}$) Similarity	(%) Target	(%) Easy	(%) Medium	(%) Hard	(%) Attacks
MPG:(1)	.163	.323	-	.144	0	.856	.856
OPG:(1)	**.710**	.475	.944	.040	.123	.837	.837
OPG:(1)+(2)	.519	.445	.917	.013	0	.987	**.987**
OPG:(1)+(2)+(3)	.619	.440	.955	.032	.021	.947	.968
OPG:(1)+(2)+(3)+(4)	.695	**.517**	**.963**	.070	.021	.909	.930

(a) Results with instruction I_2.

Generator	(%) Implicit	($\overline{x}$) Similarity	(%) Target	(%) Easy	(%) Medium	(%) Hard	(%) Attacks
MPG:(1)	.196	.344	-	.183	0	.817	.817
OPG:(1)	**.734**	.488	.935	.056	.155	.780	.818
OPG:(1)+(2)	.535	.437	.941	.017	.046	.937	.937
OPG:(1)+(2)+(3)	.641	.459	**.963**	.032	.027	.941	**.968**
OPG:(1)+(2)+(3)+(4)	.710	**.505**	.947	.095	.017	.888	.905

(b) Results with instruction I_3.

Table 5: Generation results with GPT3 using instructions I_2, I_3 and generator `OPG:(1)+(2)+(3)+(4)`.

5 Improving HS Classifiers on Implicit Messages

In this section, we report on how our generation framework can be used to improve implicit hate speech detection on the ISHate benchmark, relying on a variant of the *build it, break it, fix it* approach (Dinan et al., 2019; Vidgen et al., 2021).

5.1 Build it, Break it, Fix it Strategy

The *build it, break it, fix it* method translates a concept used in engineering to find faults in systems, to machine learning. The breaker would seek for failures in an already built classification model: the more failures the breaker can find, the better the fixes on the model might be.

During the "build it" phase, a machine learning classifier M_0 is trained on the training set $\texttt{train}_D$ of a benchmark D, which defines a certain classification task. This classifier works as the baseline model we aim to improve. During the "breaking" phase, adversarial examples that break the initial model M_0 should be created by giving one possible configuration of parameters to our generation framework. For the following round i, $i > 1$, the same generator must break the model obtained in the previous round M_{i-1}. The generator is fed with the benchmark training set $\texttt{train}_D$ to generate hard cases through demonstration examples, so that to attack the classifier of the previous step M_{i-1}. During the "fix it" phase the model M_i is updated with the newly generated adversarial data from the "break it" round. Each newly corrected model is evaluated on the test set of the benchmark. At the same time, hyperparameter selection and loss evaluation can be performed through the development set of D.

	Non-HS			Explicit HS			Implicit HS		
M	P	R	F1	P	R	F1	P	R	F1
M_0	.90	.90	**.90**	.83	.83	.83	.50	.56	.53
M_1	.93	.85	.89	.81	.84	.82	.43	.82	.56
M_2	.93	.85	.89	.81	.85	**.83**	.45	.82	**.58**
M_3	.93	.82	.87	.81	.85	.83	.36	.82	.50

Table 6: HateBERT classification results on ISHate

5.2 Experimental Settings

Goal of these experiments is to improve the classification results of the HateBERT model (described in Section 4.2) on the `Implicit HS` class, without affecting its performances on the `Non-HS` and `Explicit HS` labels. The number of rounds used is $R = 3$, where each round has a total of 870 human-validated HS implicit adversarial examples. All the instances are scored by the goal function described in Section 3.3 in order to keep only those that are considered being `Hard`. Also, as our approach generates instances of only one class (i.e., `Implicit HS`), we randomly picked the same number of safe and explicit instances from ISHate. For the training parameters for the model, we follow (Ocampo et al., 2023), i.e., a $\texttt{batch\_size} = 2$, $\texttt{epochs} = 4$, $\texttt{lr} = 2 \times 10^{-5}$, and $\texttt{weight\_decay} = 0.01$.

5.3 Obtained Results

Table 6 reports the obtained classification results of HateBERT on the ISHate test set after three retraining rounds. All runs show satisfactory performances in the three categories. We note that after the first round, M_1 had an increment on the F1-score of the implicit class compared with its predecessor M_0 giving more relevance to recall (R) than precision (P). Consider that the generator's goal function aims to produce an implicit HS message that maximizes the chances of being classified

as either explicit or non-hateful, i.e., it forces the classifier to have false negatives. In that sense, after retraining, the SOTA model focuses its attention on prioritizing more recall than precision, as we can see in the results. Also, note that the results obtained on the majority classes are not affected. This might be a consequence of not only providing adversarial examples on the target class, but also retraining on instances that are non-hateful and explicit from the ISHate dataset. After the second round M_2, we see an improvement in the performances of the implicit class. M_2 can update its parameters on this retraining round through the provided material, that helped it to correctly classify instances that were previously misclassified by M_1. On the contrary, from the third iteration, while the classification of false negatives improves through the generator's attacks, this forces the classifier to prioritize recall over precision, affecting the overall performance of M_3 on the implicit class.

5.4 Error Analysis

To gain insights on the model's behavior, we perform a manual error analysis. As mentioned in Section 5.3, most of the errors after the fixing rounds come from false positives. In particular, as HateBERT prioritizes recall over precision, explicit and non-hateful instances labeled as implicit hate started to be more prominent. Among misclassified messages, there are some instances of explicit hate which are considered implicit by HateBERT, as Example 1.

1. *I believe that same sex attraction is a mental disorder that can be cured.*

We can also find misclassified messages when commonly used explicit words are used in neutral situations. For example, the word animal, which is used sometimes in a pejorative form, might trigger the classifier to detect it as hateful (see Example 2).

2. *No, they are the fastest land animal on this continent, so I think they are simply testing their speed.*

6 Conclusions

In this paper, we have presented a new framework for generating on-scale adversarial implicit texts for HS detection using auto-regressive language models. The proposed framework follows (Hartvigsen et al., 2022) and guides a GPT3 PLM to output messages that fool current HS classifiers on implicit messages. Following (Hartvigsen et al., 2022), we have further developed a variant of constrained beam search decoding, providing a guided generation strategy through *i)* implicit hate demonstration-based prompts, *ii)* scores of implicit hate detectors, *iii)* generation weights through implicit and explicit words in HS lexicons, and *iv)* text similarity constraints that compare the used prompt and the expected output. We show how the proposed framework can produce, from a batch of generated messages, 70% of implicit HS messages, where 90% of them result to be hard adversarial cases for a competitive SOTA model on the ISHate benchmark (i.e., HateBERT).

Furthermore, we have proposed a *build it, break it, fix it*, approach that uses the adversarial examples generated by the above described framework to incrementally retrain machine learning models and improve their classification performances. We showed how adversarial generation leverages the HateBERT classification model on the ISHate dataset by improving false negative classification. While this strategy may have potential issues, such as cyclical progress, it remains a valuable approach to improve model robustness, accelerate progress, define clear objectives (Dinan et al., 2019; Kiela et al., 2021; Vidgen et al., 2021; Wallace et al., 2022), and gain a deeper understanding of models' errors as shown in our paper.

As for future work, we plan to explore how to embed implicit and subtle statements properly in representation spaces (Kim et al., 2022; Han and Tsvetkov, 2020), deciphering models for code language (Manzini et al., 2019) and provide bias mitigation strategies for social stereotypes (Sap et al., 2020).

Limitations

The main limitation of the proposed framework is its dependency on a reasonable amount of real implicit hate instances to be used as the prompting input material. Obtaining implicit and subtle messages from social media is undoubtedly a challenging and time consuming task. More importantly, another limitation lies in the fact that the proposed framework does not rely on an automatic metric to determine if the generated messages are actually implicit. Therefore, a human-in-the-loop step for validating the obtained newly generated instances is still required. Additionally, there has been mounting pressure to obtain debiased PLMs, which might

lead to the generation of less challenging examples.

Ethics Statements

This paper contains examples of HS from existing linguistic resources for HS detection and which do not reflect the authors' opinions.

While our purpose is to prevent and curate social media resources from HS, our study might still pose a potential misuse case, as our method can be employed to encourage a large language model to generate implicit and subtle instances of hate. However, we still consider that effective classifiers and new data creation/collection methods for this task are necessary to investigate and tackle implicit and subtle online hate speech on scale and prevent the spreading of this harmful content online. Our work aims at making a step towards that objective and encourages the scientific community to investigate these aspects.

Acknowledgements

This work has been supported by the French government, through the 3IA Côte d'Azur Investments in the Future project managed by the National Research Agency (ANR) with the reference number ANR- 19-P3IA-0002.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2017. Guided open vocabulary image captioning with constrained beam search. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 936–945, Copenhagen, Denmark. Association for Computational Linguistics.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Elisa Bassignana, Valerio Basile, and Viviana Patti. 2018. Hurtlex: A Multilingual Lexicon of Words to Hurt. In Elena Cabrio, Alessandro Mazzei, and Fabio Tamburini, editors, *Proceedings of the Fifth Italian Conference on Computational Linguistics CLiC-it 2018*, pages 51–56. Accademia University Press.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Rui Cao and Roy Ka-Wei Lee. 2020. HateGAN: Adversarial generative-based data augmentation for hate speech detection. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6327–6338, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Tommaso Caselli, Valerio Basile, Jelena Mitrović, and Michael Granitzer. 2021. HateBERT: Retraining BERT for abusive language detection in English. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 17–25, Online. Association for Computational Linguistics.

Tommaso Caselli, Valerio Basile, Jelena Mitrović, Inga Kartoziya, and Michael Granitzer. 2020. I feel offended, don't be abusive! implicit/explicit messages in offensive and abusive language. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 6193–6202, Marseille, France. European Language Resources Association.

Yi-Ling Chung, Elizaveta Kuzmenko, Serra Sinem Tekiroglu, and Marco Guerini. 2019. CONAN - COunter NArratives through nichesourcing: a multilingual dataset of responses to fight online hate speech. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2819–2829, Florence, Italy. Association for Computational Linguistics.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. *Proceedings of the International AAAI Conference on Web and Social Media*, 11(1):512–515. Number: 1.

Ona de Gibert, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. 2018. Hate speech dataset from a white supremacy forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 11–20, Brussels, Belgium. Association for Computational Linguistics.

Emily Dinan, Samuel Humeau, Bharath Chintagunta, and Jason Weston. 2019. Build it break it fix it for

dialogue safety: Robustness from adversarial human attack. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4537–4546, Hong Kong, China. Association for Computational Linguistics.

Mai ElSherief, Caleb Ziems, David Muchlinski, Vaishnavi Anupindi, Jordyn Seybolt, Munmun De Choudhury, and Diyi Yang. 2021. Latent hatred: A benchmark for understanding implicit hate speech. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 345–363, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Margherita Fanton, Helena Bonaldi, Serra Sinem Tekiroğlu, and Marco Guerini. 2021. Human-in-the-loop for data collection: a multi-target counter narrative dataset to fight online hate speech. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Antigoni Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large Scale Crowdsourcing and Characterization of Twitter Abusive Behavior. *Proceedings of the International AAAI Conference on Web and Social Media*, 12(1). Number: 1.

Björn Gambäck and Utpal Kumar Sikdar. 2017. Using convolutional neural networks to classify hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90, Vancouver, BC, Canada. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

Hugo Lewi Hammer. 2017. Automatic Detection of Hateful Comments in Online Discussion. In *Industrial Networks and Intelligent Systems*, pages 164–173, Cham. Springer International Publishing.

Xiaochuang Han and Yulia Tsvetkov. 2020. Fortifying toxic speech detectors against veiled toxicity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7732–7739, Online. Association for Computational Linguistics.

Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022. ToxiGen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3309–3326, Dublin, Ireland. Association for Computational Linguistics.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546, Vancouver, Canada. Association for Computational Linguistics.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4129–4142, Hong Kong, China. Association for Computational Linguistics.

David Jurgens, Libby Hemphill, and Eshwar Chandrasekharan. 2019. A just and comprehensive strategy for using NLP to address online abuse. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3658–3666, Florence, Italy. Association for Computational Linguistics.

Jonathan W. Kanter, Monnica T. Williams, Adam M. Kuczynski, Katherine E. Manbeck, Marlena Debreaux, and Daniel C. Rosen. 2017. A Preliminary Report on the Relationship Between Microaggressions Against Black People and Racism Among White College Students. *Race and Social Problems*, 9(4):291–299.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. 2021. Dynabench: Rethinking benchmarking in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4110–4124, Online. Association for Computational Linguistics.

Youngwook Kim, Shinwoo Park, and Yo-Sub Han. 2022. Generalizable implicit hate speech detection using contrastive learning. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 6667–6679, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Hannah Kirk, Bertie Vidgen, Paul Rottger, Tristan Thrush, and Scott Hale. 2022. Hatemoji: A test suite and adversarially-generated dataset for benchmarking and detecting emoji-based hate. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1352–1368, Seattle, United States. Association for Computational Linguistics.

Ju-Hyoung Lee, Jun-U Park, Jeong-Won Cha, and Yo-Sub Han. 2019. Detecting context abusiveness using hierarchical deep learning. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 10–19, Hong Kong, China. Association for Computational Linguistics.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: Adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202, Online. Association for Computational Linguistics.

Thomas Manzini, Lim Yao Chong, Alan W Black, and Yulia Tsvetkov. 2019. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 615–621, Minneapolis, Minnesota. Association for Computational Linguistics.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. TextAttack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126, Online. Association for Computational Linguistics.

Kevin L. Nadal, Katie E. Griffin, Yinglee Wong, Sahran Hamit, and Morgan Rasmus. 2014. The Impact of Racial Microaggressions on Mental Health: Counseling Implications for Clients of Color. *Journal of Counseling & Development*, 92(1):57–66. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/j.1556-6676.2014.00130.x.

Isar Nejadgholi, Kathleen Fraser, and Svetlana Kiritchenko. 2022. Improving generalizability in implicitly abusive language detection with concept activation vectors. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5517–5529, Dublin, Ireland. Association for Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Nicolas Ocampo, Ekaterina Sviridova, Elena Cabrio, and Serena Villata. 2023. An in-depth analysis of implicit and subtle hate speech messages. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1997–2013, Dubrovnik, Croatia. Association for Computational Linguistics.

Ji Ho Park and Pascale Fung. 2017. One-step and two-step classification for abusive language detection on Twitter. In *Proceedings of the First Workshop on Abusive Language Online*, pages 41–45, Vancouver, BC, Canada. Association for Computational Linguistics.

Paul Röttger, Bertie Vidgen, Dong Nguyen, Zeerak Waseem, Helen Margetts, and Janet Pierrehumbert. 2021. HateCheck: Functional tests for hate speech detection models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 41–58, Online. Association for Computational Linguistics.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. 2020. Social bias frames: Reasoning about social and power implications of language. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5477–5490, Online. Association for Computational Linguistics.

Sheikh Muhammad Sarwar and Vanessa Murdock. 2022. Unsupervised domain adaptation for hate speech detection using a data augmentation approach. *Proceedings of the International AAAI Conference on Web and Social Media*, 16(1):852–862.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2019. The woman worked as a babysitter: On biases in language generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3407–3412, Hong Kong, China. Association for Computational Linguistics.

Bertie Vidgen, Tristan Thrush, Zeerak Waseem, and Douwe Kiela. 2021. Learning from the worst: Dynamically generated datasets to improve online hate detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1667–1682, Online. Association for Computational Linguistics.

Eric Wallace, Adina Williams, Robin Jia, and Douwe Kiela. 2022. Analyzing dynamic adversarial training data in the limit. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 202–217, Dublin, Ireland. Association for Computational Linguistics.

Kunze Wang, Dong Lu, Caren Han, Siqu Long, and Josiah Poon. 2020. Detect all abuse! toward universal abusive language detection models. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6366–6376, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding abuse: A typology of abusive language detection subtasks. In *Proceedings of the First Workshop on Abusive Language Online*, pages 78–84, Vancouver, BC, Canada. Association for Computational Linguistics.

Michael Wiegand, Elisabeth Eder, and Josef Ruppenhofer. 2022. Identifying implicitly abusive remarks about identity groups using a linguistically informed approach. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5600–5612, Seattle, United States. Association for Computational Linguistics.

Michael Wiegand, Maja Geulig, and Josef Ruppenhofer. 2021a. Implicitly abusive comparisons – a new dataset and linguistic analysis. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 358–368, Online. Association for Computational Linguistics.

Michael Wiegand, Josef Ruppenhofer, and Elisabeth Eder. 2021b. Implicitly abusive language – what does it actually look like and why are we not getting there? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 576–587, Online. Association for Computational Linguistics.

Michael Wiegand, Josef Ruppenhofer, Anna Schmidt, and Clayton Greenberg. 2018. Inducing a lexicon of abusive words – a feature-based approach. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1046–1056, New Orleans, Louisiana. Association for Computational Linguistics.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section "Limitations", page 9

☑ A2. Did you discuss any potential risks of your work?
Section "Ethics Statements", page 9

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract and Section "Introduction" page 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

We use the ISHate dataset, section 4.2

☑ B1. Did you cite the creators of artifacts you used?
Section 4.2

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
The dataset is a collection of available datasets.

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Section 4.2

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
The dataset is a collection of available datasets that have properly anonymized.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Section 4.2

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 4.2

C ☑ Did you run computational experiments?

Sections 4.3 and 5.2

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Sections 4.3 and 5.2

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Sections 4.3 and 5.2

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Sections 4.4 and 5.3

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Sections 4.4 and 5.3

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.