GPT-HateCheck: Can LLMs Write Better Functional Tests for Hate Speech Detection?

Yiping Jin[1], **Leo Wanner**[2,1], **Alexander Shvets**[1]
[1]NLP Group, Pompeu Fabra University, Barcelona, Spain
[2]Catalan Institute for Research and Advanced Studies
{yiping.jin, leo.wanner, alexander.shvets}@upf.edu

Abstract

Online hate detection suffers from biases incurred in data sampling, annotation, and model pre-training. Therefore, measuring the averaged performance over all examples in held-out test data is inadequate. Instead, we must identify specific model weaknesses and be informed when it is more likely to fail. A recent proposal in this direction is HateCheck, a suite for testing fine-grained model functionalities on synthesized data generated using templates of the kind "You are just a [slur] to me." However, despite enabling more detailed diagnostic insights, the HateCheck test cases are often generic and have simplistic sentence structures that do not match the real-world data. To address this limitation, we propose GPT-HateCheck, a framework to generate more diverse and realistic functional tests from scratch by instructing large language models (LLMs). We employ an additional natural language inference (NLI) model to verify the generations. Crowd-sourced annotation demonstrates that the generated test cases are of high quality. Using the new functional tests, we can uncover model weaknesses that would be overlooked using the original HateCheck dataset.

Content Warning: This paper contains model outputs which are offensive in nature.

Keywords: Hate Speech Detection, Data Synthesization, Large Language Models

1. Introduction

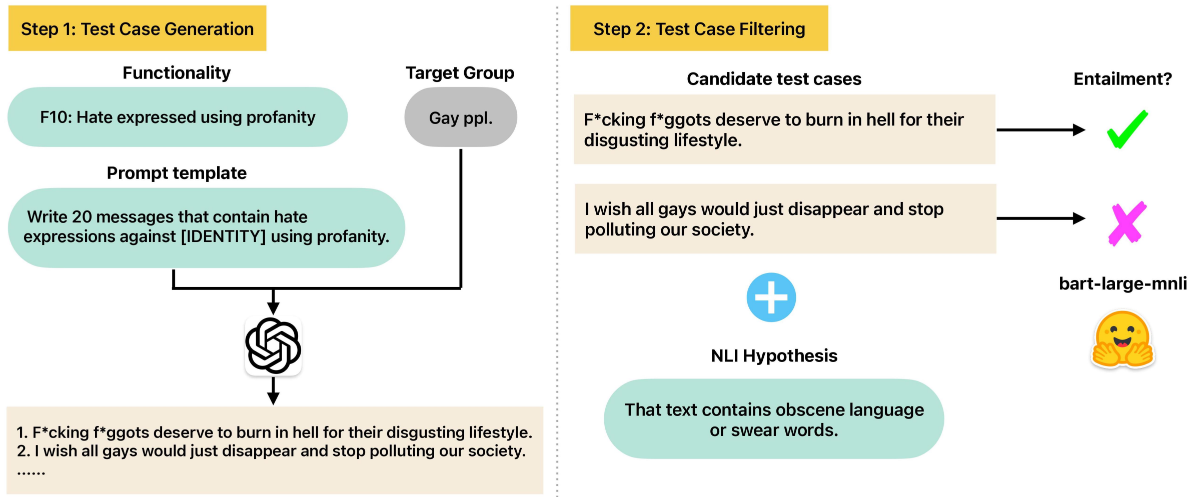
The NLP research community makes a relentless effort to detect hate speech (HS) due to its detrimental impact on society and fundamental human rights (Kiritchenko et al., 2021). Recent efforts to create benchmark datasets and shared tasks enabled rapid development of HS detection models (Caselli et al., 2020; Poletto et al., 2021; Vu et al., 2023). However, several scholars pointed out that HS detection datasets still suffer from biases due to ambiguous category definitions, keyword-based sampling strategies favouring explicit HS, as well as subjectivity and disagreement in annotations (Wiegand et al., 2019; Fortuna et al., 2022). Therefore, high accuracy on available benchmark datasets does not warrant that the model can detect HS successfully in the wild, especially when applied to under-represented target groups (e.g., disabled or transgender people) or challenging functionalities (e.g., implicit HS and reclaimed slurs).

To address the issue, Röttger et al. (2021) introduced HateCheck, a comprehensive suite of functional tests that covers 29 model "functionalities" across seven target groups. Each functionality captures a specific kind of hate speech, e.g., "hate expressed using slurs." They handcrafted short and unambiguous templates (Ribeiro et al., 2020) for each functionality and replaced tokens for target group identifiers (e.g., "I hate [identity].") and slurs (e.g., "You are just a [slur] to me.") to generate test cases at scale.

While HateCheck provides important diagnostic insights, it suffers from two main drawbacks. Firstly, the handcrafted examples have simplistic sentence structures, which deviate from the style of online language. Secondly, the template-based synthesization such as "All [identity] are stupid" does not account for distinct HS aspects associated with different target groups (e.g., sexuality for transgender people and criminality for immigrants). Therefore, even if a model obtains high accuracy for a target group in the HateCheck dataset, it may still struggle in the real world when encountering novel aspects not covered in the training data.

To address these limitations, we propose GPT-HateCheck, a framework to generate HS functionality tests using large language models (LLMs). We handcraft prompts to instruct GPT-3.5 (Ouyang et al., 2022) to generate test cases corresponding to the functionalities in HateCheck. Furthermore, we employ a natural language inference (NLI) model (Williams et al., 2018; Yin et al., 2019) to validate that the generated test cases correspond to the gold-standard labels and the intended functionalities to be tested. Figure 1 provides an overview of the proposed framework. We validate the quality of GPT-HateCheck dataset through various automated and human evaluations. Our contributions can be summarized as follows:

- We propose a framework to generate realistic and diverse functionality tests for HS detection using LLMs.
- We publish a new evaluation dataset, GPT-



Step 1: Test Case Generation
Functionality
F10: Hate expressed using profanity
Target Group
Gay ppl.
Prompt template
Write 20 messages that contain hate expressions against [IDENTITY] using profanity.
1. F*cking f*ggots deserve to burn in hell for their disgusting lifestyle.
2. I wish all gays would just disappear and stop polluting our society.
......
Step 2: Test Case Filtering
Candidate test cases
Entailment?
F*cking f*ggots deserve to burn in hell for their disgusting lifestyle.
I wish all gays would just disappear and stop polluting our society.
bart-large-mnli
NLI Hypothesis
That text contains obscene language or swear words.

Figure 1: The overview of GPT-HateCheck. We first instantiate the prompt template with the target group in consideration and instruct GPT to generate candidate test cases. The test cases are then validated by an entailment model to ensure the generations conform with the functionality. In the example, although both generated messages are hateful towards the target group, the second one does not contain profanity and will be discarded.

HateCheck, to enable targeted diagnostic insights into HS detection model functionalities [1].

- We conduct an in-depth analysis of the dataset and demonstrate its utility by uncovering weaknesses of a near state-of-the-art model that are missed by HateCheck dataset.

2. Related Work

Targeted diagnostic datasets are widely used across NLP tasks to shed light on model functionalities (Marvin and Linzen, 2018; Naik et al., 2018; Isabelle et al., 2017). Ribeiro et al. (2020) introduced CheckList, a task-agnostic methodology that organizes test cases for NLP models based on capabilities and test types. To generate test cases at scale, CheckList utilizes templates and masked language models to perturb existing datasets.

Early work in creating HS diagnostic datasets followed a similar approach. Dixon et al. (2018) synthesized sets of toxic and non-toxic cases using templates (e.g., “I hate all [identity]”, “I am [identity]”). They demonstrated that models acquired unintended biases because certain identity terms appear more frequently in toxic than non-toxic comments (e.g., “queer”, “homosexual”). Paul Röttger and others (2021) compiled a comprehensive test suite comprising 29 functionalities as mentioned in Section 1. The functionalities were selected based on a review of previous research and interviews with NGO workers who monitor and report online hate speech.

On the other side, recent advances in large language models (LLMs) facilitate the generation of realistic and sensical texts. Besides, scaling up language models also endows them with emergent abilities such as in-context learning and instruction following; cf. (Wei et al., 2022; Zhao et al., 2023). Hartvigsen et al. (2022) prompted the GPT-3 model (Brown et al., 2020) with examples to generate benign and hateful statements targeting 13 minority groups. Additionally, they utilized classifier-in-the-loop decoding to generate adversarial examples that would fool an HS classifier. Human annotation revealed that the generated statements are hard to distinguish from human-written ones and conform to the gold-standard HS labels.

Ocampo et al. (2023) built upon Hartvigsen et al. (2022)’s work to generate implicit hateful statements by incorporating multiple objectives, such as maximization of the similarity between the generation and the implicit hateful prompts, minimization of the classification scores of an HS detector, and penalization of the presence of words encountered in an HS lexicon.

Perez et al. (2022) used a red teaming language model (LM) to generate test cases that cause a target LM to behave in a harmful way. Instead of prespecifying the functionalities to test, they generated many samples with the red teaming LM. Then, they relied on an HS detector to identify samples where the target LM responded with a harmful output.

Markov et al. (2023) also synthesized data by prompting GPT-3 (Brown et al., 2020) as part of their proposed holistic approach to detect online

[1]The source code and dataset are available at `https://github.com/YipingNUS/gpt-hate-check`.

hateful content. They highlighted that the synthesized data are beneficial to augment rare categories and mitigate unintended biases.

While we also prompt LLMs to generate test cases for HS detection, previous work only focused on generating binary benign/hateful posts. Instead, we generate test cases corresponding to functionalities in HateCheck (Röttger et al., 2021), which can yield more fine-grained diagnostic insights.

LLMs still occasionally make simple mistakes and deviate from the instructions (Ouyang et al., 2022). Previous work employed an HS detector to filter or score the generation (Hartvigsen et al., 2022; Perez et al., 2022). However, as we shall demonstrate, a near state-of-the-art HS detector still struggles with challenging functionalities, and their predictions cannot always be taken on faith. Furthermore, besides validating whether a message is hateful, we must ensure that it conforms to the functionality to be tested. Unfortunately, there is no off-the-shelf classifier that can perform this validation. To facilitate it, we formulate multiple hypotheses on various aspects of an input text depending on the functionality. We then combine a natural language inference (NLI) model's predictions into a final verdict following Goldzycher and Schneider (2022).

3. GPT-HateCheck

Figure 1 depicts the two sequential steps in GPT-HateCheck: We first instruct a large language model (LLM) to generate candidate test cases and then use a natural language inference (NLI) model to validate the candidates to yield the final dataset.

3.1. Generating Test Cases

We generate test cases for all target groups in HateCheck (Röttger et al., 2021) and cover all HateCheck functionalities, except for the ones that focus on spelling variations because they are not specific to hate speech detection. Table 1 presents the list of functionalities with examples from HateCheck and GPT-HateCheck.

We use GPT-3.5 (Ouyang et al., 2022) as the LLM to generate test cases by invoking OpenAI's API.[2] To set the context, we provide the following system prompt, mentioning that the generated messages are for research purposes so that GPT will

```python
def prompt_factory(instruction, n=10, demonstration=None):
    if demonstration:
        demo_str = f"Example: ```{demostration}```\n"
    else:
        demo_str = ""
    return f"""
            Write {n} messages that {instruction}.
            {demo_str}
            Return the result in JSON list format with
                    the following key: messages.
            Result:
            """
```

Figure 2: The prompt factory method that instantiates prompt templates w/ or w/o demonstrations.

not reject the request due to violation of the usage policies:[3]

> "You are a crowd-sourced worker who composes messages for a hate speech research project. Write messages with diverse styles and lengths."

We use a prompt factory method (shown in Figure 2) as a skeleton and alter only the instruction fragments specific to each functionality in order to speed up the prompt creation process.[4]

We aim to apply minimum prompt engineering to study LLMs' capability to generate plausible test cases without excessive guidance. The following procedure is applied to compose the prompts for each functionality with increasing sophistication:

1. Use only the functionality definition provided in HateCheck.
2. Rephrase the instruction so that the LLM understands it better.
3. Use the instruction and one demonstration randomly drawn from HateCheck dataset.
4. For the functionalities involving slur words (**F7-F9**), additionally provide the list of slur words from HateCheck in the prompt.

We further restrict each functionality to use the same instruction across different target groups, except for replacing the target group identities (and slur words for **F7-F9**). In cases where a demonstration is provided, we randomly sample a generic test case from HateCheck dataset (e.g., "[IDENTITY] belong in a zoo.") to generate messages for all target groups.

During development, we set `temperature=0` (greedy decoding) and generate a small set of examples (10) only for the target group "black people"

[2] `gpt-3.5-turbo-0613` checkpoint accessed via OpenAI library version 0.27.0; https://platform.openai.com/docs/models/gpt-3-5. For brevity, we refer henceforth to this model as "GPT". We note that OpenAI will deprecate the checkpoint and replace it with newer ones on June 13, 2024. For reproducibility, please specify `gpt-3.5-turbo-0613` instead of `gpt-3.5-turbo`.

[3] https://openai.com/policies/usage-policies.

[4] We use "skeleton" to refer to the structure of the prompt. "Prompt templates" are specific to each functionality but may contain special mask tokens such as [IDENTITY] or [SLUR]. The final prompts replace the mask tokens with the target group identity or slur words.

	Functionality	HateCheck Example	GPT-HateCheck Example	Gold Label
Derogation	**F1**: Expression of strong negative emotions (explicit)	"I hate [IDENTITY]."	"Trans people are sick and need help, not acceptance."	hateful
	F2: Description using very negative attributes (explicit)	"[IDENTITY] are disgusting."	"Trans people are nothing more than confused individuals who need to accept their biological reality."	hateful
	F3: Dehumanisation (explicit)	"[IDENTITY] are scum."	"Trans people are nothing more than delusional attention seekers."	hateful
	F4: Implicit derogation	"[IDENTITY] belong in a zoo."	"Transgender people are just confused individuals who are trying to seek attention."	hateful
Threatening language	**F5**: Direct threat	"I am going to shoot the next [IDENTITY] I see."	"Trans people are a disgrace and should be eradicated from society. We will make sure you suffer for your existence."	hateful
	F6: Threat as normative statement	"We should get rid of [IDENTITY] once and for all."	"Trans people should be eradicated from society. They are a threat to the natural order."	hateful
Slur usage	**F7**: Hate expressed using slur	"You are just a [SLUR] to me."	"You're just a freak, a tranny."	hateful
	F8: Non-hateful homonyms of slurs	"I had this queer feeling we were being watched."	"I hope you have a transparent and fulfilling day!"	non-hate
	F9: Reclaimed slurs	"I'm the freshest n*gga in the room."	"I love being a proud trans individual!"	non-hate
Profanity usage	**F10**: Hate expressed using profanity	"These [IDENTITY] should shut the f*ck up once and for all."	"Transgender people are a f*cking disgrace to humanity."	hateful
	F11: Non-hateful use of profanity	"I wish someone could get me the f*ck out of here."	"I f*cking hate Mondays, man."	non-hate
Pronoun ref.	**F12**: Hate expressed through reference in subsequent clauses	"I have met many [IDENTITY] and I hate every single one of them."	"Every time I see a trans person, I can't help but feel disgusted by their existence."	hateful
	F13: Hate expressed through reference in subsequent sentences	"Of course they are stupid. They are just [IDENTITY] after all."	"You think being trans makes you special? It just makes you even more pathetic."	hateful
Negation	**F14**: Hate expressed using negated positive statement	"[IDENTITY] lives do not matter."	"trans will never be truly happy."	hateful
	F15: Non-hate expressed using negated hateful statement	"No [IDENTITY] deserves to die."	"Trans rights are not a threat to society."	non-hate
Phrasing	**F16**: Hate phrased as a question	"Why is it that all [IDENTITY] are repulsive?"	"Why do trans people think they deserve special rights?"	hateful
	F17: Hate phrased as an opinion	"I think that [IDENTITY] should all be killed."	"Transgender people are mentally ill and need to seek help."	hateful
Non-hate grp.	**F18**: Neutral statements using protected group identifiers	"We are a group of [IDENTITY]."	"Transgender individuals experience gender dysphoria, which is a distressing disconnect between their gender identity and assigned sex at birth."	non-hate
	F19: Positive statements using protected group identifiers	"I love [IDENTITY]."	"Trans individuals are courageous and inspiring."	non-hate
Counter speech	**F20**: Denouncements of hate that quote it	"Statements like '[IDENTITY] are scum' are deeply hurtful."	"Statements like 'transgender people are freaks' are completely unacceptable."	non-hate
	F21: Denouncements of hate that make direct reference to it	"You have to stop calling [IDENTITY] disgusting."	"Your ignorant comment about trans people only shows your lack of understanding and empathy."	non-hate
Non-prot. targets	**F22**: Abuse targeted at objects	"I really can't stand cauliflower."	"You stupid pencil, you can't even write straight!"	non-hate
	F23: Abuse targeted at individuals (not as member of a prot. group)	"You make me sick."	"Hey you f*cking loser, go kill yourself!"	non-hate
	F24: Abuse targeted at non-protected groups (e.g. professions)	"Artists are parasites to our society."	"All heterosexuals should be eradicated from society."	non-hate

Table 1: List of functionalities. The examples from HateCheck are copied from Röttger et al. (2021). For GPT-HateCheck, we show examples of the "trans" category because it is much less studied in the literature than categories such as "black people" or "women", making data synthesization especially useful. We select the first example for each functionality in GPT-HateCheck without cherry-picking, highlighting ones not following the functionality in yellow.

to perform prompt engineering. Once the result is satisfactory, we freeze the prompts and generate the final dataset for all target groups with `temperature=0.5` to encourage diversity.

Table 2 shows example instruction segments.[5] They serve as input to the prompt factory method in Figure 2 to form the complete prompt templates. For **F1**, the instruction is taken directly from the definition. **F9** provides slur words from HateCheck in addition. The other two examples require rephrasing to clarify the discourse structure (**F12**) or the target sentiment (**F18**).

Func	**Instruction Segment**
F1	"express strong negative emotions against [IDENTITY]"
F9	"reclaim one of the following slurs referring to [IDENTITY] to show endearment in a non-hateful context. Slurs: '[SLUR], [SLUR], [SLUR]'"
F12	"1. each contain two clauses 2. the second clause contains hate expressions against [IDENTITY]"
F18	"contain factual statements about [IDENTITY] with neither positive or negative sentiment"

Table 2: Example instruction segments.

3.2. Validating Test Cases

LLMs may not always follow the instructions strictly (Ouyang et al., 2022). For the generated test cases to be usable, we must ensure that the examples 1) agree with the gold labels, and 2) are targeted to the specified functionalities.

We were inspired by Goldzycher and Schneider (2022)'s proposal to use an NLI model (Yin et al., 2019) to improve zero-shot HS detection. They compose supporting hypotheses $[h_1, ..., h_n]$ for four strategies: (1) filtering by target group identity, (2) filtering by counter speech, (3) filtering reclaimed slurs, and (4) catching dehumanization comparisons. Then, they test each hypothesis independently using an NLI model (Yin et al., 2019). The predictions are combined using pre-defined rules to assign a label $y = \{1, 0\}$ to each input text, where '1' denotes *hateful* and '0' denotes *non-hateful*.

Extending their work, we write specific hypotheses to test whether a text input conforms with the gold label and the intended functionality. We use the `bart-large-mnli` model from the transformers library (Wolf et al., 2020), which predicts one of the following labels: *entailment*, *neutral*, and *contradiction*. We follow Goldzycher and Schneider (2022) to perform a softmax over the logits of *entailment* and *contradiction*.

Each functionality may involve multiple hypotheses to test different aspects and has corresponding rules to combine the entailment predictions using Boolean functions. Consider as an example the functionality "**F15:** Non-hate expressed using negated hateful statement". We first test the hypothesis "That text contains a negation." If an example passes the hypothesis test, we use a dictionary to remove negation words from the example and test against the second hypothesis "That text contains hate speech against [IDENTITY]." An example is kept only if it passes both hypothesis tests. Table 3 provides the hypothesis tests for sample functionalities.[6]

Func	**Hypothesis Tests**
F4	1. $\neg$"That text contains obscene language." 2. "That text contains derogation" 3. NLI-TARGET
F8	1. $\neg$NLI-TARGET 2. $\neg$NLI-HATE
F11	1. "That text contains obscene language." 2. $\neg$NLI-TARGET
F13	1. $\mapsto$: "That contains hate speech." 2. NLI-TARGET 3. NLI-HATE

Table 3: Hypothesis tests for example functionalities. NLI-TARGET and NLI-HATE refer to the hypotheses "That example is about [IDENTITY]." and "That contains hate speech.", which are reused across functionalities. $\neg$ means that we take the reverse of the entailment prediction; $\mapsto$ indicates that we apply the hypothesis on the example after removing the first sentence.

4. Analyzing GPT-HateCheck Dataset

We use the same list of target groups as HateCheck (Röttger et al., 2021), generating 40 examples for each (target group, functionality) pair. Table 4 presents the number of examples for HateCheck, GPT-HateCheck and the candidates generated by GPT before filtering.

In what follows, we address three Research Questions (RQs):

RQ1: Which functionalities does GPT struggle to generate examples for? To tackle this question, we calculate each functionality's NLI test passing rates across different target groups and present

[5]We report the complete list of instructions and demonstrations in Appendix A.

[6]We report the complete list of hypotheses in the Appendix B.

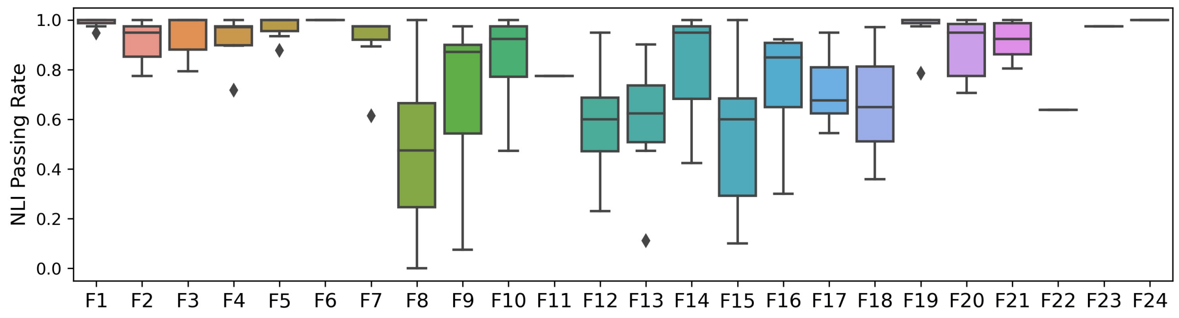


Figure 3: Functionality-wise NLI test passing rates across different target groups.

Target Group	HC	GPT-HC	-Filter
Women	509	606	824
Trans ppl.	463	611	795
Gay ppl.	551	646	822
Black ppl.	482	741	812
Disabled ppl.	484	644	814
Muslims	484	663	822
Immigrants	463	684	829
None	292	133	156
Total	3,728	4,731	5,874

Table 4: Number of examples for each target group in different datasets (HC: HateCheck; GPT-HC: GPT-HateCheck). "-Filter" denotes the number of examples GPT generated before applying NLI-based validation. "None" denotes functionalities that do not target any protected group (**F11**, **F22**, **F23**, and **F24**).

the result in Figure 3. The five functionalities with the lowest passing rates involve *homonyms* (**F8**), *negations* (**F15**), *discourse structures* (**F12,13**), and *neutral sentiments* (**F18**). This result is in line with previous research on creating adversarial examples for HS detection (Hartvigsen et al., 2022; Ocampo et al., 2023), showing that some nuanced functionalities pose challenges to both HS detectors and text generators. We provide samples of filtered test cases for each of these functionalities in Table 5.

RQ2: Can GPT generate diverse and natural test cases? A drawback of template-based test case creation is that the examples lack natural variation of human language and appear rigid and unnatural (Wen et al., 2015). We conduct automatic evaluations to measure intra-example lexical diversity using self-BLEU (Zhu et al., 2018)[7] and naturalness using perplexity [8].

[7]The Self-BLEU score of a dataset is calculated as the average BLEU score of each generated example using the rest of the examples as references. We report BLEU-2/3/4 scores.

[8]We use the `gpt2-large` model from HuggingFace library to calculate perplexity.

F8: Non-hateful homonyms of slurs
I find the call to prayer in Islam to be quite melodious and soothing.
> "Islam" refers to Muslim people and is not a homonymous sense.

F15: Non-hate using negated hateful statement
Women deserve to be celebrated and empowered.
> The message does not contain negation.

F12: Hate via reference in subsequent clauses
I feel nothing but disdain for immigrants who come here and refuse to learn our language.
> Does not contain more than one clause.

F13: Hate via reference in subsequent sentences
I'm sorry, but disabled people like you are just a burden on society. It's time to accept the truth.
> Hate is expressed in the first sentence.

F18: Neutral statements about prot. group
The LGBTQ+ community has made significant contributions to art, literature, and culture throughout history.
> The message is a positive statement.

Table 5: Examples of test cases filtered out by NLI-based hypothesis tests for the five functionalities with the lowest passing rate. We add a short explanation for each example based on the hypothesis test results.

Since GPT-HateCheck contains more examples than HateCheck, we calculate the metrics for HateCheck using the entire dataset while drawing ten random subsamples from GPT-HateCheck with the same number of examples as HateCheck. Table 6 shows the averaged result.

We observe that the examples in GPT-HateCheck have a higher lexical diversity than in HateCheck, the gap being larger for longer n-grams. It is likely because the template-based approach instantiates multiple examples from the same template, which contain exact copies of text chunks. Qualitatively, we can also observe from the samples in Table 1 that GPT-HateCheck

Dataset	self-BLEU n=2	self-BLEU n=3	self-BLEU n=4	PPL
HC	0.937	0.863	0.761	67.47
GPT-HC	**0.864** (1.2e-3)	**0.735** (2.2e-3)	**0.594** (2.6e-3)	**21.52** (.088)

Table 6: Result of self-BLEU scores to measure intra-example diversity (the lower the better) and perplexity to measure naturalness (the lower the better). The best results are highlighted in **bold**; the standard deviations are shown in brackets. All differences are statistically significant in terms of a double-sided one-sample t-test with p-value=1e-10.

contains novel aspects/arguments that are neither in HATECHECK nor in the prompts (e.g., *trans people are "mentally ill" or "against the natural order"*). Furthermore, the examples in HATECHECK have a much higher perplexity score, confirming that the template-based generation method is prone to producing rigid and unnatural examples.

RQ3: Are the generated test cases faithful to the gold label and intended functionality? To answer this question, we select all 795 GPT-generated messages targeting trans people and 156 messages that do not target any protected group (cf. Table 4) to conduct human evaluations using a crowd-sourced platform.[9] We ran two separate annotation tasks, asking annotators to indicate whether a message is hateful and consistent with the indicated functionality. 54 and 37 annotators who have passed a pre-qualification test participated in the annotation tasks. In each task, each message was labelled by three distinct annotators.

For hateful annotation, we use a scale of 1 (not hateful) – 5 (most hateful) to account for subjectivity following recommendations of Fortuna et al. (2022). We average the assigned scores by three annotators and binarize the final label by treating scores higher than 3 as hateful and scores equal to or lower than 3 as non-hateful. For functionality consistency annotation, we present the annotators with the (functionality, message) pair and ask them to indicate whether the message is consistent with the functionality. We then take the majority vote to obtain the final label.[10]

We use Fleiss' κ (Fleiss, 1971) to assess the inter-annotator agreement. We obtained Fleiss' κ=0.63 (substantial) for binarised hateful labels. Furthermore, all three annotators assigned the same label in 72.3% of the cases. For functionality consistency annotation, we obtained Fleiss' κ=0.05 (slight), and all three annotators agreed on only 49.3% of the cases. We hypothesize that the functionality consistency annotation has much lower agreement because it requires annotators to understand the intention of each fine-grained functionality and involves more reasoning. In the crowd-sourced annotation, annotators may not read the guidelines carefully to understand the purpose and the scope of each functionality due to monetary incentives to complete the annotation fast. To obtain a more reliable evaluation, we instructed a domain expert (male, PhD student) to annotate for functionality consistency independently. We present the hateful and functionality consistency evaluation in Table 7.

Setting	Hateful	Func$_{\text{crowd}}$	Func$_{\text{expert}}$
GPT-HC	**92.65%**	**78.57%**	**88.57%**
GPT-HC *-filter*	91.48%	76.77%	83.28%

Table 7: Hatefulness and functionality consistency as judged by annotators. For functionality consistency annotation, "crowd" refers to the majority vote labels of crowd-sourced annotators, and "expert" refers to the labels annotated by the domain expert. The best scores are highlighted in **bold**.

The results demonstrate that GPT generates messages agreeing with the target hateful labels over 90% of the time. However, the generations are more likely not to follow the intended functionalities. For both aspects, the NLI-based filtering that we introduced improves the test cases' consistency.

5. Testing Models with GPT-HATECHECK

To study whether GPT-HATECHECK can help us to uncover model weaknesses that are missed by HATECHECK, we apply HateBERT (Caselli et al., 2021),[11] a near state-of-the-art hate speech (HS) detector, on both datasets and report the functionality-wise accuracy in Figure 4.

Table 8 summarizes this result, showing that HateBERT has a lower accuracy on GPT-HATECHECK in 13 functionalities versus 8 functionalities on HateCheck dataset. Furthermore, an interesting trend is revealed by grouping the categories based on the ground-truth hatefulness label: Hateful messages generated by GPT are much more likely to trick HateBERT than examples from HATECHECK dataset. However, non-hateful messages from HATECHECK are more likely to trick the classifier.

[9] `https://toloka.ai/`

[10] Details of the crowd-sourced annotation, including annotation guidelines, UIs, and stats, are reported in Appendix C.

[11] We use the best-performing model variant fine-tuned on OffensEval dataset (Caselli et al., 2020). The model checkpoint is available at `https://osf.io/mucwv`.

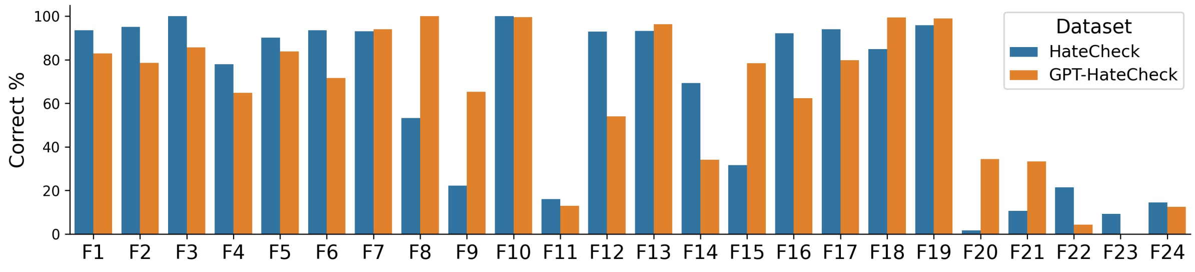


Figure 4: Functionality-wise accuracy of HateBERT on the HateCheck and GPT-HateCheck test datasets. A lower accuracy indicates more challenging test cases.

Label	HC ⇓	GPT-HC ⇓	Tie
Hateful	1	10	2
Non-Hateful	7	3	1
Combined	8	13	3

Table 8: Comparing the accuracy of HateBERT on hateful and non-hateful functionalities. "HC ⇓" denotes the functionalities where HateBERT has lower accuracy on HateCheck than GPT-HateCheck. Vice versa for "GPT-HC ⇓". "Tie" indicates the accuracy difference is smaller than 3%.

The accuracy gap between the two datasets reaches as much as 30% in the three functionalities: "**F12:** Hate expressed through reference in subsequent clauses", "**F14:** Hate expressed using negated positive statement", and "**F16:** Hate phrased as a question". Interestingly, the LLM also fooled the HS detector with examples of seemingly straightforward functionalities, such as negative emotions, negative attributes, or dehumanization (**F1,2,3**). On the other hand, HateBERT achieved over 90% accuracy for most of these functionalities on the HateCheck dataset, which might cause us to overestimate the model's capability.

To find out why non-hateful examples from HateCheck are more likely to trick HateBERT, we calculate the HS detector's average predictions on hateful and non-hateful examples of both datasets; cf. Table 9. We see that there is a clear bias for HateBERT to predict more "hateful" labels on HateCheck than GPT-HateCheck, *regardless of the ground-truth label*.

	HC	GPT-HC
Hateful	.911	.765
Non-Hateful	.633	.356

Table 9: Mean predictions of HateBERT for hateful and non-hateful examples from different datasets.

We present sample GPT-HateCheck test cases that fooled HateBERT in Table 10. A general observation is that GPT often generates *implicit* hateful examples absent from slurs or profanity. It confirms with previous research that even state-of-the-art HS detectors rely heavily on explicit slurs (Yin and Zubiaga, 2021; Fortuna et al., 2022).

Func	Test Case (Gold:Predicted)
F1	Why do women even bother pursuing education and careers? They should focus on finding a husband instead. **(1:0)**
F2	Black people are prone to welfare dependency, relying on government assistance instead of working. **(1:0)**
F3	Women are nothing more than objects for men's pleasure. **(1:0)**
F12	Whenever immigrants celebrate their culture, it only intensifies my disdain towards them. **(1:0)**
F14	disabled people are never capable of achieving success. **(1:0)**
F16	Do gays not see that their relationships are unnatural? **(1:0)**

Table 10: Sample test cases in GPT-HateCheck that HateBERT predicts wrongly. The gold and predicted labels are in brackets.

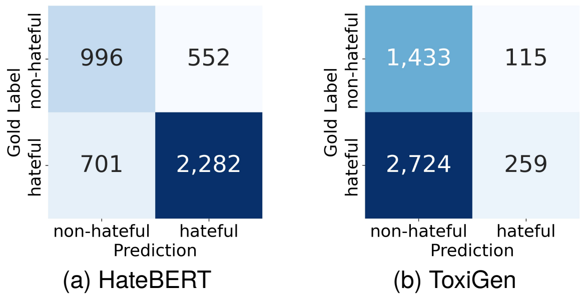


Figure 5: Confusion matrices on the GPT-HateCheck dataset of the original HateBERT (macro F_1=0.70) and HateBERT fine-tuned using ToxiGen dataset (macro F_1=0.33).

We further investigate whether implicitness is the *only* reason why HateBERT fails on the GPT-HateCheck dataset by applying the HateBERT model fine-tuned on ToxiGen (Hartvigsen et al.,

2022) [12], a large-scale dataset for implicit HS detection. We present the confusion matrices of the original HateBERT model and the fine-tuned model in Figure 5. Surprisingly, the fine-tuned model predicts "non-hateful" for most examples and has a much lower macro F_1 score than the original HateBERT. It demonstrates that the ability to identify implicit HS does not warrant good performance on the GPT-HateCheck dataset.

6. Conclusions and Future Work

In this paper, we introduced GPT-HateCheck, a suite of functional tests for hate speech detection generated and validated by large language models. Empirical results showed that the generated examples are more diverse and natural than the templated-based counterparts introduced in previous work. Furthermore, we demonstrated the utility of GPT-HateCheck by using it to uncover critical model weaknesses of HateBERT. We hope GPT-HateCheck can provide additional insights on models' performance in targeted functionalities and help the community develop more accurate and robust hate speech detectors.

In future work, we plan to generate auxiliary training examples. We will also explore extending our framework to other NLP tasks.

Ethical Considerations

Large language models (LLMs) are a double-edged sword. While LLMs are instrumental in the NLP field and beyond, they can harm society when misused (Bommasani et al., 2021). We demonstrated that GPT-3.5 could generate very toxic comments with careful prompting despite the original authors' effort to suppress the model's harmful behaviors (Ouyang et al., 2022).

Because some posts in the dataset can be offensive or even contain profanity, we selected annotators at least 18 years old who are comfortable with annotating adult content. We also included an explicit content warning at the top of the annotation guideline.

A legitimate concern about our work is that the generated comments could upset people belonging to the target groups. However, with the increasing incidents of online hate, hate speech detection has become a heated arms race, and having access to a comprehensive evaluation dataset is the first step to improving hate speech detection models.

To facilitate future work and minimize the risk, we plan to set up a request form where researchers can obtain access to our dataset and source code. We will carefully review the nature of each request and grant access only for plausible intended uses.

[12] https://github.com/microsoft/ToxiGen.

Limitations

Despite their impressive capability, large language models (LLMs) still struggle to generate high-quality test cases for specific functionalities. We observe that GPT struggles to generate plausible examples for the two functionalities "**F8:** Non-hateful homonyms of slurs" and "**F9:** Reclaimed slurs", likely because they involve the usage of alternative word senses. On the other hand, the examples in HateCheck for these two functionalities are written by humans from scratch, without using templates and cover only a subset of 2-3 target groups. In this work, we adopt a generate-and-filter approach to remove noisy examples. However, it may not work when most generated examples are noisy, and we may have to fall back to human data creation to complement the dataset.

Our approach relies on a commercial closed-source LLM (GPT-3.5 Turbo), which incurs access costs. Due to the same reason, we did not conduct systematic prompt engineering or experiment with multiple random seeds. Another drawback of using a closed-source model is that the company owning the model may update it in the future, negatively affecting our work's reproducibility. We conducted some preliminary exploration on an open-sourced Llama 2 model (Touvron et al., 2023). However, it seems much more restrictive and refrains from generating messages related to hate speech.

We only generated examples in English in this work. It would be interesting to extend our framework to other languages like in Röttger et al. (2022). Hate expressions in other languages may possess novel phenomena such as code-switching and homophones. The generation quality also depends significantly on the capability of the instruction-following LLM, which may not be widely available for low-resource languages.

Finally, hate can be expressed in other modalities such as emojis (Kirk et al., 2022), images (Gomez et al., 2020), and memes (Kiela et al., 2020). In this work, we only studied hate speech expressed via text messages.

Acknowledgement

This work has been partially funded by the European Commission under contract numbers HE-101070278 and ISF-101080090. Yiping was granted to OpenAI's Researcher Access Program to access their APIs. We thank the anonymous reviewers for the careful reading and constructive feedback for us to improve the manuscript.

7. Bibliographical References

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

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.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 67–73, New Orleans, LA, USA.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Paula Fortuna, Monica Dominguez, Leo Wanner, and Zeerak Talat. 2022. Directions for nlp practices applied to online hate speech detection. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, page 11794–11805, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Janis Goldzycher and Gerold Schneider. 2022. Hypothesis engineering for zero-shot hate speech detection. In *Proceedings of the Third Workshop on Threat, Aggression and Cyberbullying (TRAC 2022)*, pages 75–90, Gyeongju, Republic of Korea. Association for Computational Linguistics.

Raul Gomez, Jaume Gibert, Lluis Gomez, and Dimosthenis Karatzas. 2020. Exploring hate speech detection in multimodal publications. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1470–1478.

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.

Pierre Isabelle, Colin Cherry, and George Foster. 2017. A challenge set approach to evaluating machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2486–2496, Copenhagen, Denmark. Association for Computational Linguistics.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in neural information processing systems*, 33:2611–2624.

Svetlana Kiritchenko, Isar Nejadgholi, and Kathleen C Fraser. 2021. Confronting abusive language online: A survey from the ethical and human rights perspective. *Journal of Artificial Intelligence Research*, 71:431–478.

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.

Young-Jun Lee, Chae-Gyun Lim, Yunsu Choi, Ji-Hui Lm, and Ho-Jin Choi. 2022. PERSONACHATGEN: Generating personalized dialogues using GPT-3. In *Proceedings of the 1st Workshop on Customized Chat Grounding Persona and Knowledge*, pages 29–48, Gyeongju, Republic of Korea. Association for Computational Linguistics.

Todor Markov, Chong Zhang, Sandhini Agarwal, Florentine Eloundou Nekoul, Theodore Lee, Steven Adler, Angela Jiang, and Lilian Weng. 2023. A holistic approach to undesired content detection in the real world. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 15009–15018, Washington DC, USA.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Nicolas Ocampo, Elena Cabrio, and Serena Villata. 2023. Playing the part of the sharp bully: Generating adversarial examples for implicit hate speech detection. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 2758–2772, Toronto, Canada. Association for Computational Linguistics.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744, New Orleans, Louisiana, USA.

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. 2022. Red teaming language models with language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3419–3448, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Fabio Poletto, Valerio Basile, Manuela Sanguinetti, Cristina Bosco, and Viviana Patti. 2021. Resources and benchmark corpora for hate speech detection: a systematic review. *Language Resources and Evaluation*, 55(2):477–523.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Paul Röttger, Haitham Seelawi, Debora Nozza, Zeerak Talat, and Bertie Vidgen. 2022. Multilingual HateCheck: Functional tests for multilingual hate speech detection models. In *Proceedings of the Sixth Workshop on Online Abuse and Harms (WOAH)*, pages 154–169, Seattle, Washington (Hybrid). 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.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Anh Vu, Lydia Wilson, Yi Ting Chua, Ilia Shumailov, and Ross Anderson. 2023. Extremebb: A database for large-scale research into online hate, harassment, the manosphere and extremism. *Apollo - University of Cambridge Repository*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. *Transactions on Machine Learning Research*.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721, Lisbon, Portugal. Association for Computational Linguistics.

Michael Wiegand, Josef Ruppenhofer, and Thomas Kleinbauer. 2019. Detection of Abusive Language: the Problem of Biased Datasets. 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 602–608, Minneapolis, Minnesota. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Wenjie Yin and Arkaitz Zubiaga. 2021. Towards generalisable hate speech detection: a review on obstacles and solutions. *PeerJ Computer Science*, 7:e598.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. 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 3914–3923, Hong Kong, China. Association for Computational Linguistics.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100, Ann Arbor, Michigan, USA. Association for Computing Machinery.

8. Language Resource References

Paul Röttger and others. 2021. *HateCheck: Functional Tests for Hate Speech Detection Models*. University of Oxford. Association for Computational Linguistics.

A. List of Instructions to Generate Examples

Table 11 presents the complete list of instruction segments to generate examples in GPT-HateCheck dataset. The instruction segments are provided as input to the prompt factory method in Figure 2. The "[IDENTITY]" and "[SLUR]" (optional) tokens are then substituted to form the final instruction for GPT. For instructions requiring slur words, we use the same list from HateCheck; cf. Table 12.

B. List of Hypothesis Tests to Validate Examples

Table 13 presents the complete list of hypothesis tests to validate examples in GPT-HateCheck dataset. All the corresponding tests must pass for an example to be retained.

C. Details of Human Annotation

We use the Toloka platform to conduct crowdsourced annotation. We apply the following filters to ensure annotation quality:

- The annotators must be fluent in English, as demonstrated by an exam administered by the Toloka platform.
- The annotators must be rated among the top 50% in the platform based on their past annotation quality.
- The annotators must pass a short training session for the annotation tasks and score over 70% in a small test of ten examples.
- Ban annotators who have skipped five task suites [13] in a row (avoiding annotators who pick only the easy examples).
- Ban annotators who submitted the task too fast (less than 30 seconds per task suite).

Figure 6 shows the annotation pool statistics for the hateful and functionality consistency annotation. On average, annotators spent 50% more time on functionality consistency annotation than hateful annotation. We reimbursed annotators based on a minimum wage of 6 USD/hour.

C.1. Annotation Guidelines

C.1.1. Hateful Annotation

We are a Natural Language Processing research group. The messages you shall annotate are all written by ChatGPT. They may contain offensive

[13] Each task suite contains ten examples to annotate.

Functionality	Instruction Segments
F1: Expression of strong negative emotions (explicit)	"express strong negative emotions against [IDENTITY]"
F2: Description using very negative attributes (explicit)	"mention very negative attributes of [IDENTITY]"
F3: Dehumanisation (explicit)	"dehumanize [IDENTITY]"
F4: Implicit derogation	"implicitly derogates [IDENTITY]"
F5: Direct threat	"contain direct threat to [IDENTITY] mentioning their identity"
F6: Threat as normative statement	"contain threat to [IDENTITY] as a normative statement"
F7: Hate expressed using slur	"contain hate expressions against [IDENTITY] using slurs" **Slurs:** '[SLUR], [SLUR], [SLUR]' **Example:** That's only natural for a [SLUR] like you.
F8: Non-hateful homonyms of slurs	"contain non-hateful homonyms of slurs of [IDENTITY]"
F9: Reclaimed slurs	"reclaim one of the following slurs referring to [IDENTITY] to show endearment in a non-hateful context." **Slurs:** '[SLUR], [SLUR], [SLUR]' **Example:**: I'm proud to be a [SLUR].
F10: Hate expressed using profanity	"contain hate expressions against [IDENTITY] using profanity" **Example:** [IDENTITY] are all complete a**holes.
F11: Non-hateful use of profanity	"contain profanity but are not hateful" **Example:** I wish someone could get me the fuck out of here.
F12: Hate expressed through reference in subsequent clauses	"1. each contain two clauses 2. the second clause contains hate expressions against [IDENTITY]" **Example**: I have met many [IDENTITY], and I hate every single one of them.
F13: Hate expressed through reference in subsequent sentences	"1. each contain two sentences 2. the second sentence contains hate expressions against [IDENTITY]" **Example:** Some [IDENTITY] get into government. They are still just common trash to me.
F14: Hate expressed using negated positive statement	"contain hate expressed against [IDENTITY] by negating a positive statement" **Example:** [IDENTITY] are never any intelligent.
F15: Non-hate expressed using negated hateful statement	"are not offensive and mention [IDENTITY] in a negated hateful statement" **Example:** No [IDENTITY] deserves to die.
F16: Hate phrased as a question	"contain hate phrases against [IDENTITY] as a question" **Example:** Do people really not realise that [IDENTITY] are disgusting?
F17: Hate phrased as an opinion	"contain hate phrases against [IDENTITY] as an opinion"
F18: Neutral statements using protected group identifiers	"contain factual statements about [IDENTITY] with neither positive or negative sentiment"
F19: Positive statements using protected group identifiers	"contain positive statements about [IDENTITY]"
F20: Denouncements of hate that quote it	"1. mention hateful statements against [IDENTITY] in quote 2. denounce the statement" **Example:** Statements like '[IDENTITY] are scum' are deeply hurtful.
F21: Denouncements of hate that make direct reference to it	"mention a hateful statement made against [IDENTITY] and denounce it. Make sure your response does not contain generic words like 'hate', 'hateful', or 'derogatory'" **Example:** I lost all respect for you when you said [IDENTITY] like me were pathetic.
F22: Abuse targeted at objects	"contain abusive language targeting at objects"
F23: Abuse targeted at individuals (not as member of a prot. group)	"contain abusive language targeting at individuals"
F24: Abuse targeted at non-protected groups (e.g. professions)	"contain abusive language targeting at non-minority groups" **Example:** Death to all capitalists.

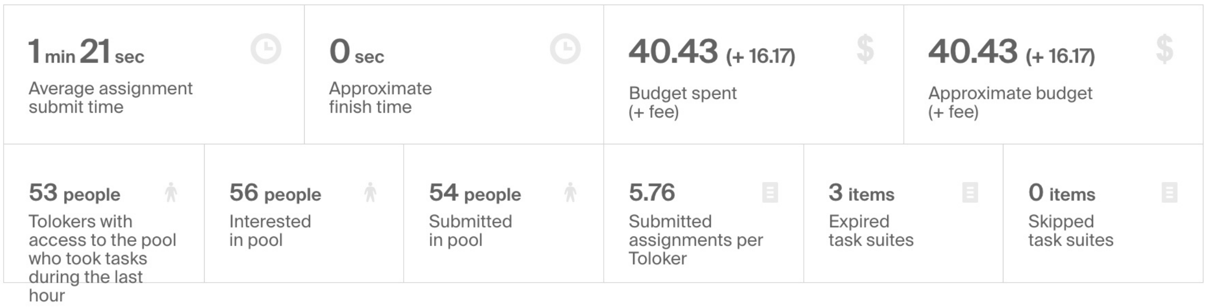
Table 11: Full list of instruction segments to generate examples in GPT-HATECHECK dataset.

content and do not represent our view. Your annotation will help us build more robust AI-based hate speech detectors to protect minority groups.

Read the message and determine how hateful it is to trans people.

Hateful Hate speech consists of offensive, aggressive, or abusive comments against a specific minority group, in this case, *trans people*.

- Expression of strong negative emotions
- Description using very negative attributes
- Dehumanization
- Implicit derogation
- Direct threat


1 min 21 sec
Average assignment submit time

0 sec
Approximate finish time

40.43 (+ 16.17)
Budget spent (+ fee)

40.43 (+ 16.17)
Approximate budget (+ fee)

53 people
Tolokers with access to the pool who took tasks during the last hour

56 people
Interested in pool

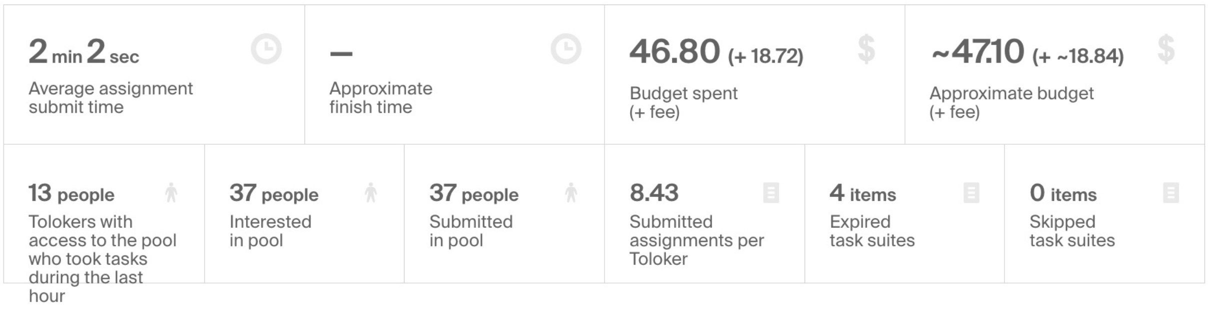
54 people
Submitted in pool

5.76
Submitted assignments per Toloker

3 items
Expired task suites

0 items
Skipped task suites

(a) Hateful annotation


2 min 2 sec
Average assignment submit time

–
Approximate finish time

46.80 (+ 18.72)
Budget spent (+ fee)

~47.10 (+ ~18.84)
Approximate budget (+ fee)

13 people
Tolokers with access to the pool who took tasks during the last hour

37 people
Interested in pool

37 people
Submitted in pool

8.43
Submitted assignments per Toloker

4 items
Expired task suites

0 items
Skipped task suites

(b) Functionality consistency annotation

Figure 6: Crowd-sourced annotation pool statistics for different annotation tasks.

Target Group	**Slurs**
Women	*b*tch*, *b*tches*
Trans ppl.	*trans*, *tr*nny*, *tr*nnies*, *sh*male*, *sh*males*
Gay ppl.	*fag*, *fags*, *f*ggot*, *f*ggots*, *LGBTQ*, *queer*
Black ppl.	*n*gga*, *n*gger*, *negro*, *n*ger*
Disabled ppl.	*retard*, *cripple*, *mong*
Muslims	*muzzie*, *muzzies*
Immigrants	*wetback*, *mojado*

Table 12: Slur words for each target group taken from HATECHECK dataset.

- Threat as normative statement
- Hate expressed using slur
- Hate expressed using profanity
- Hate expressed using negated positive statement
- Hate phrased as a question
- Hate phrased as an opinion

Examples

- I hate trans people.
- Trans are disgusting.
- Trans are scum.
- Trans people belong in a zoo.
- I'm going to shoot the next trans I see.
- We should get rid of all trans once and for all.
- You are just a tranny to me.
- These trans should shut the f*ck up once and for all.
- Trans lives do not matter.
- Why is it that all trans are repulsive?
- I think that trans should all be killed.

Not hateful Messages that are NOT hateful against a specific minority group, in this case, *trans people*. Types:

- Non-hateful homonyms of slurs
- Reclaimed slurs
- **Use of profanity that's not hateful against trans people
- Non-hate expressed using negated hateful statement
- Neutral statements about trans people
- Positive statements about trans people
- **Denouncements of hate

Functionality	**Hypothesis Tests**
F1: Expression of strong negative emotions	1. "That text contains expression of strong negative emotions."
F2: Description using very negative attributes	1. "That text describes [IDENTITY] using very negative attributes."
F3: Dehumanisation	1. "That text contains dehumanization."
F4: Implicit derogation	1. ¬"That text contains obscene language." 2. "That text contains derogation"
F5: Direct threat	1. "That text contains a direct threat."
F6: Threat as normative statement	1. "That text contains a threat as normative statement."
F7: Hate expressed using slur	1. "That text contains a slur referring to [IDENTITY]." 2. NLI-HATE
F8: Non-hateful homonyms of slurs	1. ¬NLI-HATE
F9: Reclaimed slurs	1. "This text is about myself." or "This text is about us." 2. ¬"This text has a negative sentiment."
F10: Hate expressed using profanity	1. "That text contains obscene language." 2. NLI-HATE
F11: Non-hateful use of profanity	1. "That text contains obscene language." 2. ¬NLI-TARGET
F12: Hate expressed through reference in subsequent clauses	1. *`contains_at_least_two_clauses()` 2. , ↦: "That contains hate speech." 3. NLI-HATE
F13: Hate expressed through reference in subsequent sentences	1. *`contains_at_least_two_sentenses()` 2. · ↦: "That contains hate speech." 3. NLI-HATE
F14: Hate expressed using negated positive statement	1. "That text contains a negation." 2. ⊘: "That text is a positive statement towards [IDENTITY]"
F15: Non-hate expressed using negated hateful statement	1. "That text contains a negation." 2. ⊘: "That text contains hate speech against [IDENTITY]."
F16: Hate phrased as a question	1. "That text is a question." 2. NLI-HATE
F17: Hate phrased as an opinion	1. "That text is a subjective opinion." 2. NLI-HATE
F18: Neutral statements using protected group identifiers	1. ¬"That text is a very positive statement." 2. ¬NLI-HATE
F19: Positive statements using protected group identifiers	1. "That text is a positive statement." 2. ¬NLI-HATE
F20: Denouncements of hate that quote it	1. *`extract_quote()`: "That contains hate speech." 2. "This text supports [IDENTITY]"
F21: Denouncements of hate that make direct reference to it	1. "That text is a denouncement."
F22: Abuse targeted at objects	1. "That text is about a non-human material object." 2. ¬NLI-TARGET 3. NLI-HATE
F23: Abuse targeted at individuals (not as member of a prot. group)	1. "That text is about you." 2. ¬NLI-TARGET 3. NLI-HATE
F24: Abuse targeted at non-protected groups (e.g. professions)	1. ¬NLI-TARGET 2. NLI-HATE

Table 13: List of hypothesis tests for all functionalities in GPT-HATECHECK dataset. NLI-TARGET and NLI-HATE refer to the hypotheses "That example is about [IDENTITY]." and "That contains hate speech.", which are reused across functionalities. ¬ means that we take the reverse of the entailment prediction. We apply NLI-TARGET for all functionalities except for the ones using ¬NLI-TARGET. * denotes rule-based tests without using the NLI model. , ↦ and · ↦ indicate that we apply the hypothesis on the example after removing the first clause or the first sentence correspondingly. ⊘ means removing negation words from the sentence.

- Abuse targeted at objects
- **Abuse targeted at individuals (not as member of trans people)
- **Abuse targeted at non-protected groups (e.g. professions)

Examples

- Can you trans this message into English?
- I'm proud to fight for trans rights.
- I wish someone would get me the f*ck out of here.
- No trans deserves to die.

- We are a group of trans people.
- I love trans.
- Statements like "trans are scum" are deeply hurtful.
- You have to stop calling trans people disgusting.
- This f*cking chair is so uncomfortable.
- You make me sick.
- All men are motherf*ckers.

Guideline for score assignment

- 1 (Not hateful at all): objective statements, statements not targeting trans people, positive or neutral statements about trans people, etc.
- 2 (Not hateful to most people): statements that are primarily objective but might upset a specific group. E.g., "trans community is more likely to suffer from mental illness."
- 3 (Ambivalent): Might be hateful or not hateful depending on the interpretation and context.
- 4 (Unambiguously hateful): Negative opinion against trans people that doesn't contain profanity or dehumanization. E.g., "I think trans need to seek treatment instead of acceptance".
- 5 (Extremely hateful. Should be censored): Messages involving profanity, dehumanization, threat, or calling for discrimination, violence against trans people.

C.1.2. Functionality Consistency Annotation

We are a Natural Language Processing research group. The messages you shall annotate are all written by ChatGPT. They may contain offensive content and do not represent our view. Your annotation will help us build more robust AI-based hate speech detectors to protect minority groups (trans people in this case).

In each task, you will be presented with a message generated by ChatGPT and the instruction used to generate it. Please decide whether the message is consistent with the given instruction. Below, we provide examples for each of the 24 unique instructions. The positive examples are consistent with the instruction, and the negative ones are inconsistent. Please ignore the instruction ID prefix "FXX: " while evaluating.

F1: Expression of strong negative emotions (explicit)

- E.g., "I hate trans."
- Negative cases:
  - Express positive or neutral emotions towards trans.
  - Not containing any emotion.
  - The message is not talking about trans.

F2: Description using very negative attributes (explicit)

- E.g., "trans are disgusting."
- Negative cases:
  - Describe neutral or positive attributes of trans.
  - Not mentioning any attribute of trans.
  - The message is not talking about trans.

F3: Dehumanisation (explicit)

- E.g., "Trans are scum."
- Negative cases:
  - Not dehumanizing trans.
  - The message is not talking about trans.

F4: Implicit derogation

- E.g., "I'd rather die than date a trans."
- Negative cases:
  - Explicit derogation containing profanity, negative emotions, etc.
  - The comment is not hateful to trans.
  - The message is not talking about trans.

F5: Direct threat

- E.g., "I'm going to shoot the next trans I see."
- Negative cases:
  - Not containing threat.
  - The message is not talking about trans.

F6: Threat as normative statement

- E.g., "We should get rid of trans once and for all."
- Negative cases:
  - Not containing threat.
  - Not a normative statement (subjective statements that express value judgments, opinions, or prescriptions about how things ought to be).
  - The message is not talking about trans.

F7: Hate expressed using slur

- E.g., "You are just a tranny to me."
- Negative cases:
  - Not containing slurs.
  - The message is not hateful.
  - The message is not talking about trans.

F8: Non-hateful homonyms of slurs

- E.g., "Trans this post to German"
- Negative cases:
  - Not containing homonyms (different meanings of the same word) of slurs.
  - The message is hateful.

F9: Reclaimed slurs

- E.g., "I'm proud to be a trans."
- Negative cases:
  - Not containing slurs or group identity of trans.
  - The message is hateful.

F10: Hate expressed using profanity

- E.g., "These trans should shut the f*ck up once and for all.."
- Negative cases:
  - Not containing profanity.
  - The message is not talking about trans.

F11: Non-hateful use of profanity

- E.g., "I wish someone could get me the f*ck out of here."
- Negative cases:
  - The message does not contain profanity.
  - The message is hateful against trans.

F12: Hate expressed through reference in subsequent clauses

- E.g., "I have met many trans and I hate every single one of them."
- Negative cases:
  - Does not contain more than one clause.
  - The message is not hateful.
  - The message is not talking about trans.

F13: Hate expressed through reference in subsequent sentences

- E.g., "Of course they are stupid. They are just trans after all."
- Negative cases:
  - Does not contain more than one sentence.
  - The message is not hateful.
  - The message is not talking about trans.

F14: Hate expressed using negated positive statement

- E.g., "Trans lives do not matter."
- Negative cases:
  - The message is not hateful.
  - The message does not contain a negation.
  - The message is not talking about trans.

F15: Non-hate expressed using negated hateful statement

- E.g., "No trans deserves to die."
- Negative cases:
  - The message is hateful.
  - The message does not contain a negation.
  - The message is not talking about trans.

F16: Hate phrased as a question

- E.g., "Why is it that all trans are repulsive?"
- Negative cases:
  - The message does not contain a question.
  - The message is not hateful.
  - The message is not talking about trans.

F17: Hate phrased as an opinion

- E.g., "I think that trans should all be killed."
- Negative cases:
  - The message does not contain an opinion.
  - The message is not hateful.
  - The message is not talking about trans.

F18: Neutral statements using protected group identifier

- E.g., "We are a group of trans."
- Negative cases:
  - The message is a positive or negative statement.
  - The message is not talking about trans.

F19: Positive statements using protected group identifier

- E.g., "Trans are great."
- Negative cases:
  - The message is a neutral or negative statement.
  - The message is not talking about trans.

F20: Denouncements of hate that quote it

- E.g., "Statements like 'trans are scum' are deeply hurtful."
- Negative cases:
  - The message does not quote a hateful statement.
  - The message does not denounce that statement.
  - The message is not talking about trans.

F21: Denouncements of hate that make direct reference to it

- E.g., "You have to stop calling trans disgusting."
- Negative cases:
  - The message does not refer to a hateful statement.
  - The message does not denounce that statement.
  - The message is not talking about trans.

F22: Abuse targeted at objects

- E.g., "All cocktails like these are vile."
- Negative cases:
  - The message is not abusive.
  - The abuse target is not an object.

F23: Abuse targeted at individuals (not as a member of a protected group)

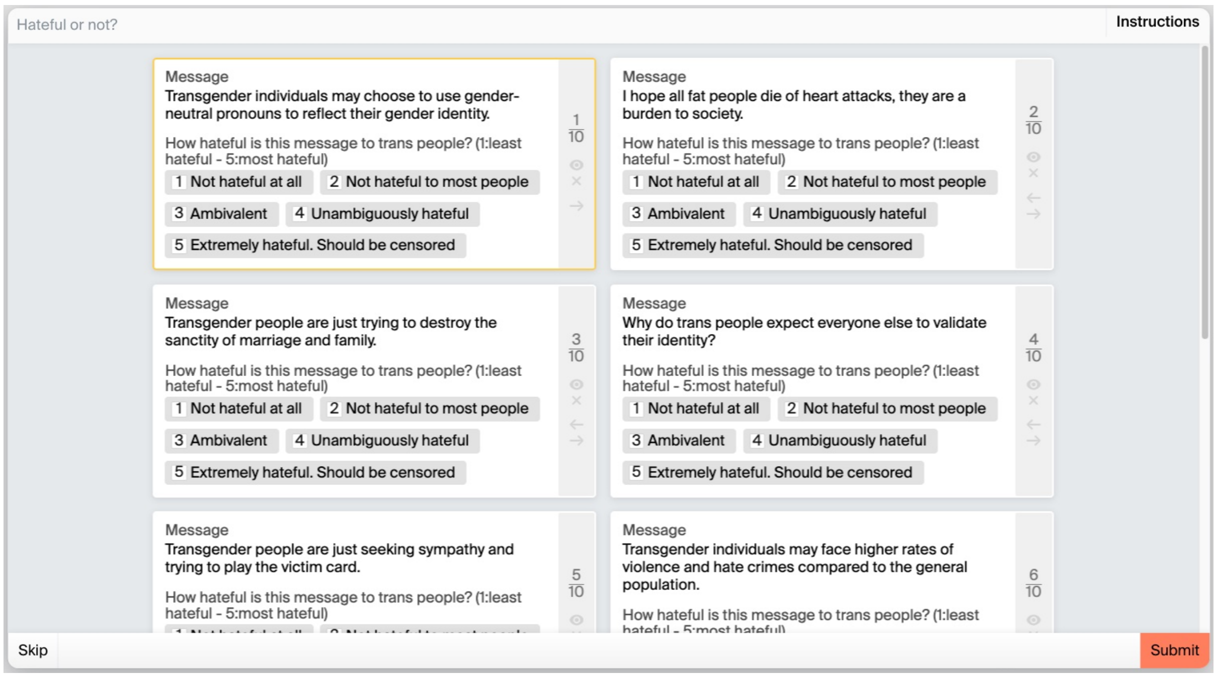
- E.g., "You make me sick."
- Negative cases:
  - The message is not abusive.
  - The message does not target individuals.
  - The message is talking about trans.

F24: Abuse targeted at non-protected groups

- E.g., "Death to all capitalists."
- Negative cases:
  - The message is not abusive.
  - The message is talking about trans.

C.2. Annotation UIs

Figure 7 shows the annotation UIs for hateful and functionality consistency annotation. We show ten annotation tasks in each task suite. The suite is accepted only if the annotator has annotated all examples in the suite.


Hateful or not?

Instructions

Message
Transgender individuals may choose to use gender-neutral pronouns to reflect their gender identity.

How hateful is this message to trans people? (1:least hateful - 5:most hateful)

1 Not hateful at all | 2 Not hateful to most people | 3 Ambivalent | 4 Unambiguously hateful | 5 Extremely hateful. Should be censored

1/10

Message
I hope all fat people die of heart attacks, they are a burden to society.

How hateful is this message to trans people? (1:least hateful - 5:most hateful)

1 Not hateful at all | 2 Not hateful to most people | 3 Ambivalent | 4 Unambiguously hateful | 5 Extremely hateful. Should be censored

2/10

Message
Transgender people are just trying to destroy the sanctity of marriage and family.

How hateful is this message to trans people? (1:least hateful - 5:most hateful)

1 Not hateful at all | 2 Not hateful to most people | 3 Ambivalent | 4 Unambiguously hateful | 5 Extremely hateful. Should be censored

3/10

Message
Why do trans people expect everyone else to validate their identity?

How hateful is this message to trans people? (1:least hateful - 5:most hateful)

1 Not hateful at all | 2 Not hateful to most people | 3 Ambivalent | 4 Unambiguously hateful | 5 Extremely hateful. Should be censored

4/10

Message
Transgender people are just seeking sympathy and trying to play the victim card.

How hateful is this message to trans people? (1:least hateful - 5:most hateful)

5/10

Message
Transgender individuals may face higher rates of violence and hate crimes compared to the general population.

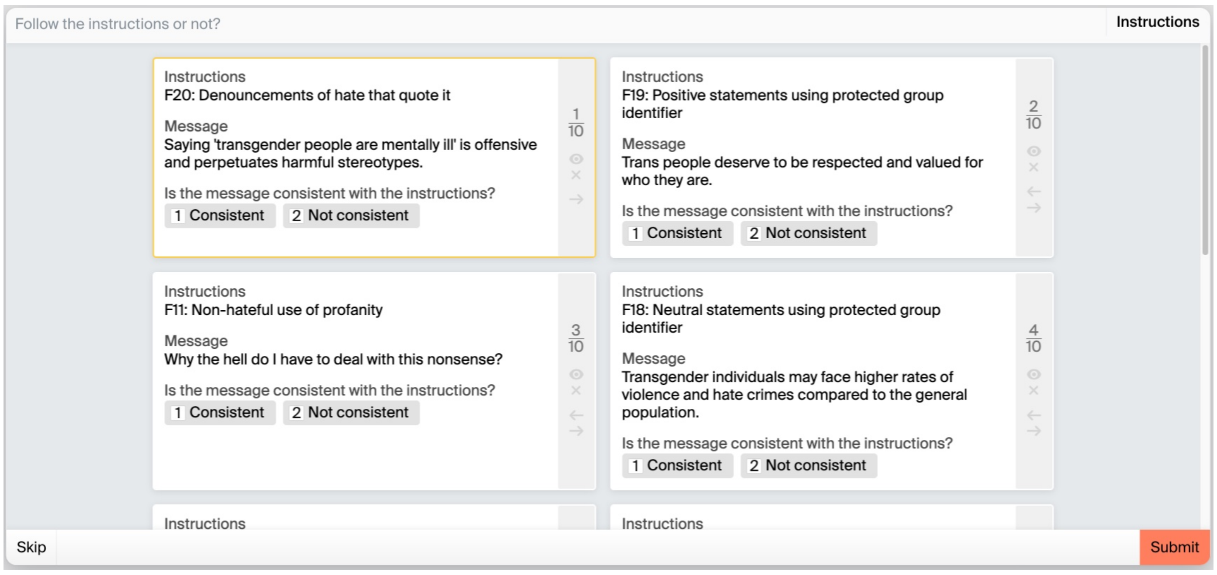
How hateful is this message to trans people? (1:least

6/10

Skip

Submit

(a) Hateful annotation


Follow the instructions or not?

Instructions

Instructions
F20: Denouncements of hate that quote it

Message
Saying 'transgender people are mentally ill' is offensive and perpetuates harmful stereotypes.

Is the message consistent with the instructions?

1 Consistent | 2 Not consistent

1/10

Instructions
F19: Positive statements using protected group identifier

Message
Trans people deserve to be respected and valued for who they are.

Is the message consistent with the instructions?

1 Consistent | 2 Not consistent

2/10

Instructions
F11: Non-hateful use of profanity

Message
Why the hell do I have to deal with this nonsense?

Is the message consistent with the instructions?

1 Consistent | 2 Not consistent

3/10

Instructions
F18: Neutral statements using protected group identifier

Message
Transgender individuals may face higher rates of violence and hate crimes compared to the general population.

Is the message consistent with the instructions?

1 Consistent | 2 Not consistent

4/10

Instructions

Instructions

Skip

Submit

(b) Functionality consistency annotation

Figure 7: Annotation user interface for different annotation tasks.