HW-TSC at Multilingual Counterspeech Generation

Xinglin Lyu*, **Haolin Wang***, **Min Zhang, Hao Yang**

Huawei Translation Services Center, Beijing, China
{lvxinglin1,wanghaolin9,zhangmin186,yanghao30}@huawei.com

Abstract

Warning: *Due to the property of the counterspeech generation (CSG) topic, this paper presents some content that may be offensive or upsetting.*

Multilingual counterspeech generation (MCSG) contributes to generating counterspeech with respectful, non-offensive information that is specific and truthful for the given hate speech, especially those for languages other than English. Generally, the training data of MCSG in low-source language is rare and hard to curate. Even with the impressive large language models (LLMs), it is a struggle to generate an appreciative counterspeech under the multilingual scenario. In this paper, we design a pipeline with a generation-reranking mode to effectively generate counterspeech under the multilingual scenario via LLM. Considering the scarcity of training data, we first utilize the training-free strategy, i.e., in-context learning (ICL), to generate the candidate counterspeechs. Then, we propose to rerank those candidate counterspeech via the Elo rating algorithm and a fine-tuned reward model. Experimental results on four languages, including English (EN), Italian (IT), Basque (EU) and Spanish (ES), our system achieves a comparative or even better performance in four metrics compared to the winner in this shared task.

1 Introduction

Hate speech (HS) refers to any form of communication that belittles or discriminates against individuals or groups based on attributes such as race, religion, ethnic origin, sexual orientation, disability, or gender (Ward, 1997; Nockleby, 2000), raising the concern to a toxic environment that affects both individuals and society as a whole (Williams, 2019). A promising countermeasure, counterspeech (CS) which response to HS that uses positive, inclusive, and factual communication to challenge harmful narratives, can effectively help address the issue while promoting a more respectful environment (Schieb and Preuss, 2016; Munger, 2017; Mathew et al., 2018; Shin and Kim, 2018), an example as shown in Table 1 Recently, numerous non-governmental organizations (NGOs) have enlisted volunteers to create counter speech by hand to address hate speech. Nevertheless, due to the overwhelming volume of hate speech online each day, automating CS generation could be more effective in reducing the need for human involvement. Therefore, increasing studies explored automating CS generation which we focus on in this paper. The one line of studies in CS generation relies on a mass of labeled data. For example, Zhu and Bhat (2021) proposes to train an RNN-based variational encoder-decoder model from scratch, to generate the multiple candidate CS and then obtain the best one from the candidate pool via pruning-selection pipeline. However, the kind of methods may be limited by the scale of labeled data. The performance of their model will significantly deteriorate when the labeled data is limited or unavailable, i.e., low-source CS generation. The other line of studies in CS generation is that integrates the powerful large language models (LLMs) when generating CS (Saha et al., 2024; Wang et al., 2024; Podolak et al., 2024). Although the impressive capacity of LLMs, generating a high-quality CS via LLMs re-

Hate Speech:	*We should ban homosexuals.*
Counter Speech:	*When will the love prosper and the hatred start to dissipate? I will not only respect my fellow LGBT+ people, I will promote their rights.*

Table 1: An example of counter speechf generation for hate speech.

*Equal contribution.

Proceedings of the First Workshop on Multilingual Counterspeech Generation (MCG), pages 47–55
January 19th, 2025.

mains a challenge due to the complexity of CS generation task, specifically for low-source language scenarios.

In this paper, we propose a generation-reranking pipeline to excavate the capacity of LLM in CS generation, specifically for the low-source language scenarios. Inspired by the success in in-context learning (ICL) (Wei et al., 2022), we first propose to inject a few of HS-CS pair examples into the prompt of LLMs. Furthermore, considering the complexity of CS generation, we employ chain-of-thought (CoT) (Wang et al., 2022; Zhang et al., 2023), a step-by-step inference mechanism, to prompt LLM to generate CS candidates. For the whole generation stage, we obtain a set of CS candidate by multiple sampling. To ensure diverse and contextually rich outputs, the generation of each CS candidate in set uses different few-shot examples that are randomly sampled from the training set.

Once the CS candidates set are obtained, we search the best CS from it via two re-ranking methods: 1) Point-wise scoring method, which introduces a point-wise scorer to independently assess each CS candidate. Each CS candidate is evaluated based on a scoring model, e.g., a reward model, and the one with the highest score is selected as the final CS output. 2) Pair-wise Comparison Method: it first pairs CS candidate in set randomly, then does a comparison for each pair of CS candidate. These pair-wise comparison results are used to compute an Elo rating for each candidate. The comparison will be performed in multiple rounds, and the later rounds will pair the CS candidate via their ELO rating, i.e., the CS candidate with higher ELO scores is more likely to be paired with the other having comparative ELO scores. This ELO-based comparison makes the ranking process more fair, and is more effective in finding the best one from the CS candidate set. Similarly, after all rounds of comparison are done, we take the CS candidate with the highest ELO rating as the final output.

Overall, our contribution can be summarized as follows:

- We propose to build a generation-reranking pipeline to effectively obtain high-quality CS from the LLM.
- We propose to combine ICL and CoT to prompt LLM and generate CS candidates during the generation stage, which can effectively overcome the complexity and data scarcity of low-source language for the CS generation task.
- We propose two re-ranking methods, which can further excavate the high-quality CS from the candidates set.
- We conduct extensive experiments and analysis, including lower-source and high-source languages, demonstrate effectiveness of our proposed approach.

2 Related Work

We introduce the related works in automatic CS generation along the following two lines: 1) generating CS via the full-training model; 2) generating CS via the pre-training model.

Generating CS via Full-Training Model. Several studies have explored generating effective counterspeech by training a model from scratch. Qian et al. (2019) train a seq2seq model over their collected dataset, then use a combination of automatic generation and human input to create CS. Hua et al. (2019) propose to integrate a retrieval model to empower the seq2seq CS generation model. Zhu and Bhat (2021) proposed an automated pipeline for generating and filtering candidate CS. Different from them, our work focus on effectively utilizing the pre-trained LLMs to obtain high-quality CS.

Generating CS via Pre-Training Model. The pre-trained models, including LLMs, have shown their powerful ability in various natural language processing (NLP) tasks. Tekiroglu et al. (2020) introduced innovative techniques for generating counterspeech with a GPT-2 model, followed by expert editing. Chung et al. (2020) examined the creation of Italian CS by fine-tuning the pre-training model. Tekiroğlu et al. (2022) further analyses several such language models and decoding strategies after fine-tuning them. Rodrguez et al. (2023) use and analyse the performance of GPT-3 in the CS generation task. Saha et al. (2024) further presents a comprehensive analysis about the CS generation capacity of various LLMs, including GPT-2 (Radford et al., 2019), DialoGPT (Zhang et al., 2020), ChatGPT (OpenAI, 2023) and FlanT5 (Chung et al., 2022). Otherwise, they also provide the different prompting strategies for generating different types of CS and analyze the impact of such strategies on the

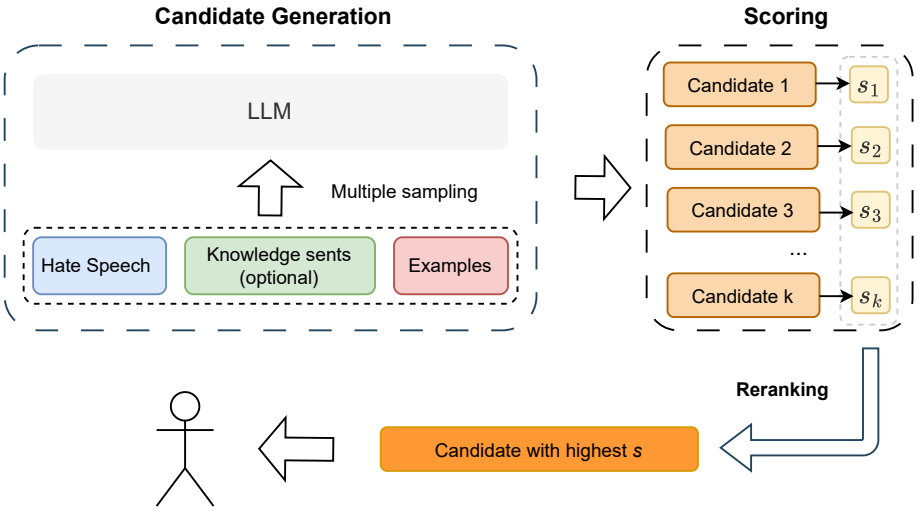

Figure 1: Overview of our proposed CS generation pipeline.

performance of the models. Different from these studies, our work is not limited CS generation of LLMs and proposes an additional re-ranking phase to more effectively mine the high-quality CS.

3 Methodology

3.1 Overview

In this section, we describe our pipeline design, as depicted visually in Fig. 2. During inference, our pipeline first utilizes LLMs to generate several candidate responses in a few-shot, chain-of-thought manner. This approach aims to ensure diverse and contextually rich outputs by leveraging effective prompt engineering techniques. Once the candidate responses are generated, we introduce two distinct methods for re-ranking them:

Point-wise Scoring Method: We use a point-wise scorer to independently assess each candidate response. Each response is graded based on a scoring model, and the response with the highest score is selected as the final output.

Pair-wise Comparison Method: We implement a pair-wise comparison mechanism. Here, pairs of candidate responses are randomly selected and compared against each other based on their relative quality. The pair-wise comparisons are used to compute Elo ratings for each candidate, and the response with the highest Elo rating is ultimately chosen.

The detailed implementation of each module in our pipeline is elaborated in the following subsections.

3.2 Candidate Generation

First, during the candidate generation phase, we adopt two common techniques used in the prompt engineering stage of large language models: In-context Learning (ICL) and Chain-of-Thought (CoT). Specifically, we follow the classic few-shot approach by selecting a number of examples from the training set to serve as the context for the LLM input. We then guide the model to reason step by step, with the ultimate goal of generating responses that mimic the style of the examples from the training set. The specific prompt template is as follows:

Regarding the number of examples used in the few-shot approach and the selection strategy, we will provide a detailed explanation and testing in the subsequent experimental section.

3.3 Point-wise Scorer Training

Without loss of generality, when introducing the method, we assume responding to a single hate speech. In training our point-wise scoring model, our goal is to predict the quality of a given counter speech response r, given the hate speech instance q, and background knowledge k. To achieve this, we first collect a dataset. Specifically, we generate k different response candidates, $[r_1, \ldots, r_k]$, for each sample in the training set. Next, we use a scoring function $\mathcal{S}(\cdot)$, to evaluate each generated response. To ensure a comprehensive evaluation,

You are tasked with counteracting hate speech. Provide counter-narrative sentences in [LANG] in response to the following hate speech statements, drawing upon the provided background knowledge sentences and hate attributes. Please ensure that you consider the following points:

1. The counter-narrative should be directly relevant to the hate speech statement.
2. Key terms from the background knowledge must be used verbatim, without modification.
3. The counter-narrative should closely mirror the expression, style, and length of the provided examples below.

Example [i]:
Hate Speech: [HS] Background Knowledge: [BK] Hate Attribute: [HA] Counter Narrative: [CN]
...

Hate Speech: [HS] Background Knowledge: [BK] Hate Attribute: [HA] Counter Narrative:

Please approach the response process systematically and outline your reasoning step by step.
- First, analyze the style and length of the provided examples.
- Second, carefully review the background knowledge and identify the key points.
- Finally, based on the guidelines outlined in the "Task Description" section, generate your final counter-speech.
The generated content should be enclosed within {TEXT}.

Figure 2: Prompt template for our CS generation.

we incorporate metrics related to the similarity to g_i. Specifically, we use RougeL, BLEU, and BertScore, which together capture both character-level and semantic-level similarity. We also introduce a high-level quality evaluation through the use of JudgeLM. JudgeLM leverages the capabilities of the language model to provide a more nuanced assessment of response quality, considering factors that may not be captured purely through similarity. The final evaluation score s_i, for each response r_i, is computed as follows:

$$\begin{aligned} s_i = \mathcal{S}(r_i) = & \, RougeL(r_i, g) + BLEU(r_i, g) \\ & + BertScore(r_i, g) \\ & + \alpha \cdot JudgeLM(r_i, g) \end{aligned} \tag{1}$$

Here, α is a weighting parameter that balances the contribution of JudgeLM relative to the similarity metrics, and g is the golden response corresponding to this hate speech. Next, to better utilize the data and improve the model's generalization capability, we adopt a Bradley-Terry style approach and train a scoring model: $\sigma(r_i; q, k)$, which is used to score each response r_i given the hate speech and background knowledge. The specific loss function is as follows:

$$\mathcal{L} = \mathbb{E}_{\mathcal{S}(r_i) > \mathcal{S}(r_j)} f(\sigma(r_i; q, k) - \sigma(r_j; q, k)) \tag{2}$$

, where $f(\cdot)$ refers to the sigmoid function.

3.4 Re-ranking Process

In the re-rank process, we select the best response from $[r_1, ..., r_k]$. Specifically, we employ two different strategies: point-wise scoring and pair-wise scoring.

For the point-wise scoring approach, we use the score model trained in the previous subsection to evaluate each response individually: $\hat{s}_i = \sigma(r_i; g, k)$. Since the input to the scoring model contains only the hate-speech, the response, and the background knowledge, we can score the responses during the testing phase without the need for a golden response. Finally, we select the response with the highest score as our final answer.

For the pair-wise scoring approach, we employ JudgeLM as an evaluator. JudgeLM is a scoring model based on a large language model that can assess pairs of text and determine the relative quality between them. We utilize this feature of JudgeLM to compute the Elo-Rating between different responses, which then serves as the basis for selecting the final response. Simply put, Elo rating serves a method to evaluate the relative quality of responses,

by continually updating their ratings based on pairwise comparisons. The Elo rating is updated using the following formula:

$$R'_A = R_A + K \cdot (S_A - E_A), \tag{3}$$

where R'_A represents the updated rating of response A,Where R'_A represents the updated rating of response A, R_A is the current rating, K is a constant that determines the sensitivity of rating changes, S_A is the actual outcome (1 if A wins, 0 otherwise), and E_A represents the expected outcome, which is calculated as follows:

$$E_A = \frac{1}{1 + 10^{(R_B - R_A)/400}} \tag{4}$$

This formula allows us to iteratively update the ratings for each response, ultimately allowing us to rank the responses based on their performance in head-to-head comparisons. The response with the highest Elo rating is selected as our final answer.

4 Experiments

4.1 Setup

Dataset Description. We utilize the dataset from the First Workshop and Shared Task on Multilingual Counterspeech Generation, designed to support the development of models for generating CS. This multilingual dataset includes instances in English, Italian, Spanish and Basque, enabling a comprehensice, multilingual evaluation on varied LLMs. Each instance comprises a HS post accompanied by background knowledge, and a manually curated golden response representing effective CS. The dataset is divided into training, validation, and test sets, with 396, 100, 100 instances in each split and each language.

LLM Generation Settings. In the default configuration, we use GPT-4o-mini to balance cost and performance. For specific generation parameters, a temperature of 1.2 and top-p of 1 are selected to enhance the diversity of generated candidates. By default, the model generates $k = 20$ responses for each HS.

Paired Dataset Collection. To construct the paired dataset for training the point-wise scorer, we generate multiple responses for each HS and score them individually based on Equation 1. Using these scores, we create 6,840 training samples and 760 test samples, which contain different hate speech instances. Each sample consists of one high-scoring response and one low-scoring response.

Point-Wise Scorer Model. For the scorer model, we adopt LLaMA-3 8B as the base model, replacing its final layer with a linear projection to predict a single scalar value. To reduce training costs, we employ the LoRA (Hu et al., 2021) method with $r = 8$ and $\alpha = 16$, and incorporate flash attention (Dao et al., 2022) to accelerate training. The training process uses the AdamW optimizer (Loshchilov, 2017) with a learning rate of 1×10^{-4}. We train the model with a batch size of 16 for a single epoch.

Metrics. We utilize the official evaluation metrics for our experimental setup, which are categorized into two types. The first type focuses on assessing the similarity between our generated responses and the reference CS across semantic, lexical, and stylistic dimensions. These metrics include RougeL (Lin, 2004), BLEU (Papineni et al., 2002), BertScore (Zhang* et al., 2020), and Novelty (Wang and Wan, 2018). The second type harnesses the capabilities of large language models (LLMs) to automatically evaluate the quality of our CS, exemplified by JudgeLM (Zhu et al., 2023). The first type effectively measures whether the generated responses leverage training data to produce style-consistent and human-like CS. However, it has limitations, particularly in its inability to holistically assess whether the CS is sharp and comprehensive. On the other hand, the second type allows for a more global evaluation but is susceptible to biases inherent in language models, such as favoring responses generated by similar models or preferring longer responses (Hu et al., 2024; Chen et al., 2024). To address these individual shortcomings, we integrate both types of evaluation criteria, achieving a more comprehensive and balanced assessment of CS quality.

4.2 Main Results

We present the performance of our proposed methods in Table 2. **Winner** and **Reference** indicate the CS generation metrics for the best-performing team and the human-annotated golden responses, respectively. I-C represents generating CS with Integrated Chain-of-Thought (ICL and CoT) but without the re-ranking phase. I-C + PwC and I-C + PwS denote the final CS output obtained through Pair-wise Comparison and Point-wise Scoring re-ranking methods, respectively. Here are some key

System	JudgeLM	RougeL	BLEU	BertScore	Length	Novelty
			English CS generation task			
Winner	2523.0	19.0	4.9	70.8	84.7	83.0
Our submission1 (I-C)	1635.0	40.4	27.2	78.2	38.2	80.7
Our submission2 (I-C + PwC)	2087.5	33.6	18.8	76.1	48.3	80.8
Our submission3 (I-C + PwS)	1682.0	46.6	34.4	80.4	39.2	79.0
Reference	1175.5	100.0	100.0	100.0	32.7	77.7
			Basque CS generation task			
Winner	2465.5	8.2	1.5	66.4	67.5	86.8
Our submission1 (I-C)	1484.5	18.3	6.3	72.1	30.2	87.2
Our submission2 (I-C + PwC)	1881.5	17.7	5.6	72.4	34.5	86.8
Our submission3 (I-C + PwS)	1722.0	23.3	10.5	74.2	32.1	86.5
Reference	1534.5	100.0	100.0	100.0	26.5	85.3
			Italian CS generation task			
Winner	1985.5	21.1	8.9	72.6	101.4	82.1
Our submission1 (I-C)	1260.5	36.1	21.7	77.2	40.8	80.9
Our submission2 (I-C + PwC)	1792.0	30.8	16.6	75.9	49.5	80.3
Our submission3 (I-C + PwS)	1372.5	41.1	26.6	79.1	41.9	79.1
Reference	929.5	100.0	100.0	100.0	35.3	77.9
			Spanish CS generation task			
Winner	2002.0	24.2	8.9	73.5	99.3	79.6
Our submission1 (I-C)	1228.5	36.8	21.7	77.6	43.1	77.5
Our submission2 (I-C + PwC)	1728.0	33.5	17.7	76.7	52.3	77.4
Our submission3 (I-C + PwS)	1339.5	41.9	27.2	79.4	43.2	75.8
Reference	899.0	100.0	100.0	100.0	36.9	75.1

Table 2: Performance of *Our submissions, Winner, Reference* on test set with four languages. **I-C** denotes generating the CS only integrates ICL and CoT without the re-ranking phase. **I-C + PwC** and **I-C + PwS** denote we obtain the final CS output via Pair-wise Comparison and Point-wise Scoring re-ranking methods, respectively.

observations:

Our method consistently achieves competitive performance across all metrics. As mentioned in the previous subsection, optimizing CS generation requires considering multiple metrics simultaneously. Our results demonstrate that our design effectively addresses this challenge. Notably, compared to the Winner, all our submissions show significant improvements in RougeL, BLEU, and BertScore. For example, the I-C + PwS submission outperforms the Winner by +27.6, +29.5, and +9.6 in these three metrics for the English CS generation task. This suggests that our ICL and CoT techniques effectively prompt the LLM to generate CS in a style that more closely resembles human outputs. Additionally, our method remains highly competitive in the JudgeLM metric, further demonstrating its overall effectiveness.

Our method are less likely to overfit to the judge model. As previously discussed, longer responses may exploit the judge model and result in inflated scores. However, our method effectively controls the response length while maintaining quality. For instance, the average length of the I-C submission is 38.2, whereas the Winner's submission averages 84.7, which may contribute to the slight gap in the JudgeLM metric. Importantly, the average length of our submissions closely aligns with the human reference, further indicating that our submissions favor a human-like CS style.

PwC substantially improves the JudgeLM metric. In the PwC method, we continuously selected CS pairs and used JudgeLM to compare and rate them via Elo rating. The results indicate

Selection Strategy	RougeL	BLEU	BertScore	Length	Novelty
Random	27.2	14.9	74.2	39.3	80.5
BertScore-based	26.6	14.0	74.3	40.1	80.7
Similarity-based	26.2	13.7	74.2	40.4	80.4

Table 3: Comparision of performance for various example selection strategies. Notably, the performance here is based on the 20 shots.

Selection Strategy	RougeL	BLEU	BertScore	Length	Novelty
I-C (*20-shot*)	27.2	14.9	74.2	39.3	80.5
1-shot	18.6	5.9	71.1	41.9	81.1
5-shot	22.6	9.8	72.7	41.1	80.9
10-shot	25.4	12.6	73.7	40.3	80.4
30-shot	27.1	14.4	74.2	39.3	80.1
50-shot	26.8	14.1	73.7	39.9	79.7

Table 4: The comparison of performance when using various numbers of examples to perform in-context learning.

Selection Strategy	RougeL	BLEU	BertScore	Length	Novelty
I-C	27.2	14.9	74.2	39.3	80.5
w/o CoT	25.5	13.0	72.3	38.1	79.8

Table 5: The ablation study of Chain-of-thought. Notably, the performance here is based on the 20 shots.

that JudgeLM's comparisons are consistent, and the selected responses effectively maximize the JudgeLM score. Comparing I-C and I-C + PwC, we observe that the latter achieves a significantly higher JudgeLM score (e.g., +452.5 points for English). However, we also note a slight decline in performance on other metrics, supporting our assertion that CS generation involves multi-objective optimization, where improving one metric may lead to trade-offs in others.

PwS enhances all metrics simultaneously. To avoid scenarios where improving some metrics results in declines in others, we designed a point-wise scorer that integrates multiple metrics, as shown in 1. Our findings show that I-C + PwS consistently improves upon I-C across all metrics. Therefore, we believe that this method has the potential to identify the Pareto optimal solution for this multi-objective optimization problem.

4.3 Ablation Study and Discussion

Here we use the merged development set (with the four language subsets), to discuss and analyze our proposed approach.

Example Selection Strategy. During the ICL phase, we incorporate some examples as a part of input to prompt the LLM generates CS with a similar style to human beings. As previous literature demonstrates, the performance of ICL is highly related to the examples chosen (Lu et al., 2022). Thus, we analyze different strategies for example selection to pursue better CS results. We evaluate 3 example selection strategies: 1) Randomly select examples in the training dataset. 2) Select examples whose hate speech (HS) has a higher BERTScore compared to the HS of input. 3) Select examples whose hate speech (HS) has a higher semantic similarity compared to the HS of input, which is measured by Jina LM (Sturua et al., 2024). We list their performances in Table 3. The results show that using similarity or BERTScore based methods can be not useful compared to a simple random strategy. This is probably because HS generation needs more diverse contexts for better generation, while previous methods may limit the context's diversity instead.

Effect of Numbers of Examples. The number of examples is also crucial for the few-shot strategy. Under-number examples may prevent the model's

ICL capabilities from being fully utilized, while over-number examples could introduce extra information burden, resulting in negative results. We compare the performance of ICL under different example numbers and the results is shown in Tale 4. We can observe that using 20 examples can maximizes the model's ICL capabilities.

Effect of Chain-of-Thought. We discuss the effect of applying Chain-of-Thought here. As shown in Table 5, the CoT can greatly improve the quality of CS in all metric such as + 1.9 Bertscore. This suggest the CoT is an effective strategy to enhance CS generation, by breaking down the complex generation process into several easier steps.

5 Conclusion

In this paper, we propose an effective pipeline for automatically generating multilingual counter-speech (MCSG) to combat hate speech with respectful and truthful responses, particularly in non-English languages. Due to the scarcity of training data for low-resource languages, we propose a pipeline that combines generation and reranking. More specifically, the proposed approach uses in-context learning (ICL) to create candidate responses without extensive training data via the powerful LLMs. These candidates are then reranked using the Elo rating algorithm and a fine-tuned reward model. The experimental results show that our system performs comparably or better than the best entry in the shared task across four languages: English, Italian, Basque, and Spanish.

References

Guiming Hardy Chen, Shunian Chen, Ziche Liu, Feng Jiang, and Benyou Wang. 2024. Humans or llms as the judge? a study on judgement biases. *arXiv preprint arXiv:2402.10669*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models.

Yi-Ling Chung, Serra Sinem Tekiroglu, and Marco Guerini. 2020. Italian counter narrative generation to fight online hate speech. In *Seventh Italian Conference on Computational Linguistics*.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Zhengyu Hu, Linxin Song, Jieyu Zhang, Zheyuan Xiao, Jingang Wang, Zhenyu Chen, and Hui Xiong. 2024. Rethinking llm-based preference evaluation. *arXiv preprint arXiv:2407.01085*.

Xinyu Hua, Zhe Hu, and Lu Wang. 2019. Argument generation with retrieval, planning, and realization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2661–2672, Florence, Italy. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*.

I Loshchilov. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. 2022. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. *arXiv preprint arXiv:2209.14610*.

Binny Mathew, Navish Kumar, Pawan Goyal, Animesh Mukherjee, et al. 2018. Analyzing the hate and counter speech accounts on twitter. *arXiv preprint arXiv:1812.02712*.

Kevin Munger. 2017. Tweetment effects on the tweeted: Experimentally reducing racist harassment. *Political Behavior*, 39(3):629–649.

John T Nockleby. 2000. Hate speech. *Encyclopedia of the American Constitution*, 3(2):1277–1279.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Association for Computational Linguistics (ACL)*.

Jakub Podolak, Szymon Łukasik, Paweł Balawender, Jan Ossowski, Jan Piotrowski, Katarzyna Bakowicz, and Piotr Sankowski. 2024. LLM generated responses to mitigate the impact of hate speech. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 15860–15876, Miami, Florida, USA. Association for Computational Linguistics.

Jing Qian, Anna Bethke, Yinyin Liu, Elizabeth Belding, and William Yang Wang. 2019. A benchmark dataset for learning to intervene in online hate speech. 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 4755–4764.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

María Estrella Vallecillo Rodrguez, Arturo Montejo-Ráez, and María Teresa Martín Valdivia. 2023. Automatic counter-narrative generation for hate speech in spanish. *Proces. del Leng. Natural*, 71:227–245.

Punyajoy Saha, Aalok Agrawal, Abhik Jana, Chris Biemann, and Animesh Mukherjee. 2024. On zero-shot counterspeech generation by LLMs. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 12443–12454, Torino, Italia. ELRA and ICCL.

Carla Schieb and Mike Preuss. 2016. Governing hate speech by means of counterspeech on facebook. In *66th ICA Annual Conference*.

Youngsoo Shin and Jinwoo Kim. 2018. Data-centered persuasion: Nudging user's prosocial behavior and designing social innovation. *Computers in Human Behavior*, 80:168–178.

Saba Sturua, Isabelle Mohr, Mohammad Kalim Akram, Michael Günther, Bo Wang, Markus Krimmel, Feng Wang, Georgios Mastrapas, Andreas Koukounas, Nan Wang, et al. 2024. jina-embeddings-v3: Multilingual embeddings with task lora. *arXiv preprint arXiv:2409.10173*.

Serra Sinem Tekiroğlu, Helena Bonaldi, Margherita Fanton, and Marco Guerini. 2022. Using pre-trained language models for producing counter narratives against hate speech: a comparative study. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3099–3114.

Serra Sinem Tekiroglu, Yi-Ling Chung, and Marco Guerini. 2020. Generating counter narratives against online hate speech: Data and strategies. In *Proceedings of ACL*.

Boshi Wang, Xiang Deng, and Huan Sun. 2022. Iteratively prompt pre-trained language models for chain of thought. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 2714–2730. Association for Computational Linguistics.

Haiyang Wang, Zhiliang Tian, Xin Song, Yue Zhang, Yuchen Pan, Hongkui Tu, Minlie Huang, and Bin Zhou. 2024. Intent-aware and hate-mitigating counterspeech generation via dual-discriminator guided LLMs. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 9131–9142, Torino, Italia. ELRA and ICCL.

Ke Wang and Xiaojun Wan. 2018. Sentigan: Generating sentimental texts via mixture adversarial networks. In *IJCAI*, pages 4446–4452.

Kenneth D Ward. 1997. Free speech and the development of liberal virtues: An examination of the controversies involving flag-burning and hate speech. *U. Miami L. Rev.*, 52:733.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022.

Matthew Williams. 2019. Hatred behind the screens: A report on the rise of online hate speech.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and William B Dolan. 2020. Dialogpt: Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2023. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Lianghui Zhu, Xinggang Wang, and Xinlong Wang. 2023. Judgelm: Fine-tuned large language models are scalable judges.

Wanzheng Zhu and Suma Bhat. 2021. Generate, prune, select: A pipeline for counterspeech generation against online hate speech. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 134–149.