Mitigating Toxicity in Dialogue Agents through Adversarial Reinforcement Learning

Guillermo Villate-Castillo[1,*], Borja Sanz[2] and Javier Del Ser[1,3]

[1]TECNALIA, Basque Research and Technology Alliance (BRTA), Derio, 48160 Bizkaia, Spain.
[2]Faculty of Engineering, University of Deusto, Avenida de las Universidades 24, 48007 Bilbao.
[3]University of the Basque Country (UPV/EHU), 48013 Bilbao, Spain.

Abstract

Large Language Models (LLMs) have revolutionized dialogue agents, but they still suffer from biases, inconsistencies, and factual inaccuracies. This paper focuses on addressing toxicity, a critical aspect of the "Diversity, non-discrimination, and fairness" pillar of Trustworthy AI, in dialogue agents. We propose a methodology inspired by InstructGPT and ChatGPT to mitigate toxicity in chatbots by incorporating toxicity detection tools from industry leaders, such as Microsoft and Google Jigsaw, into a reward model. The reward model was extended by our developed ToxDialogDefender, a context-aware toxic language identification model. To evaluate our approach, we curate a dataset of 1.5 million comments, with 14.13% serving as successful adversarial examples, to induce toxicity in the BlenderBot 1 90M model. While our primary focus is on BlenderBot 1, our approach is applicable to models with similar Seq2Seq architectures. Experimental results demonstrate a substantial reduction in toxicity levels from 24% to 5%, as validated by a subset analysis. This research highlights the potential for integrating toxicity mitigation techniques into the training paradigm of dialogue agents, paving the way for more more aligned and unbiased conversational AI systems.

Keywords

Toxicity, Alignment, Large Language Models, Reinforcement Learning

1. Introduction

Dialogue agents driven by open-domain chatbots [1, 2] play a pivotal role in applications like restaurant reservations [3], healthcare [4] and online shopping [5]. More recent cases of general-purpose dialog agents are ChatGPT [6] or Llama 2 [7], which have been trained to follow societal norms. These models undergo training with extensive datasets from platforms like Reddit[1], Twitter (currently X[2]), and 4chan[3], with examples including BlenderBot 1 [1], TwitterBot Tay [8], and Luda [9]. However, these data sources are known for producing toxic content [10, 11, 12], leading to undesirable behaviors observed in the output of these models. Toxicity mitigation is a key task at a time when the research community is fervently engaged in AI alignment and in ensuring that AI adopts human principles such as respect, fairness, non-discrimination, etc [13].

This research focuses on mitigating toxic speech in dialogue agents, which has been defined repeatedly *as rude, disrespectful, or unreasonable comments likely to disrupt conversations*[4], often related to gender, politics, race, or culture [14]. Previous efforts aimed at reducing toxicity in dialogue agents include continuous curation of datasets [15, 16], toxic behavior detection during text generation [17, 18], and safety layers [1, 19]. While effective, these approaches have limitations (L):

- L_1: The continuous curation of datasets is expensive, requiring human annotators at every stage. In

AEQUITAS 2024: Workshop on Fairness and Bias in AI | co-located with ECAI 2024, Santiago de Compostela, Spain
*Corresponding author.
guillermo.villate@tecnalia.com (G. Villate-Castillo); borja.sanz@deusto.es (B. Sanz); javier.delser@tecnalia.com (J. Del Ser)
0009-0001-0783-7984 (G. Villate-Castillo); 0000-0003-2039-7773 (B. Sanz); 0000-0002-1260-9775 (J. Del Ser)

[1]Reddit:https://www.reddit.com/
[2]X:https://twitter.com/
[3]4chan:https://www.4chan.org/index.php
[4]Perspective API, About the API - Attributes and Languages, https://developers.perspectiveapi.com/s/about-the-api-attributes-and-languages?language=en_US, accessed on April 30th, 2024.

addition, removing toxic comments may lead to the model generating unrealistic responses due to the consequent shortage of training data.

- L_2: Current toxic content detectors do not take into account the conversation history at training time, thus lacking contextualization.
- L_3: Safety layers mitigate toxicity during inference, but do not entirely eliminate it from the model's internal knowledge base. Moreover, toxicity detectors, known for their biases, can introduce unwanted biases [20]. Additionally, since next token probabilities are conditioned on the toxic detector, this may lead to incoherent responses [21].

The aforementioned limitations of current methodologies lie in three distinctive areas: adversarial training data gathering (L_1); contextualization of comments to mitigate false positives and negatives in context-sensitive comments (L_2); and intrinsic removal of toxicity from model weights (L_3).

Motivation and Research Questions To address chatbot toxicity limitations, we explore the following research questions (RQ) in corresponding order of the limitations exposed above:

- RQ_1: Are there any existing queries that drive chatbots to respond in a toxic manner?
- RQ_2: How well do toxicity detectors perform in dialog contexts?
- RQ_3: Can we eliminate toxic traits within the model without adding complexity to its architecture?

Main contributions In this work we propose a novel methodology for mitigating toxicity in dialog agent-based models. This methodology addresses the three forms in which toxicity can manifest in a discussion: implicit toxicity, explicit toxicity, and toxicity detected within the dialog context. To the best of our knowledge, this work is the first to utilize such a unique approach in dialog settings based on our literature review. Additionally, our proposed dialog context-based toxicity detector is designed to assist in situations where isolated comments are insufficient for assessing the toxicity level, particularly in cases where the model responds affirmatively to toxic questions or statements. Furthermore, we expand the pool of adversarial examples introduced in [22] for BlenderBot 1 by analyzing an additional 1.5 million examples. Finally, by leveraging Reinforcement Learning (RL), we are able to mitigate toxicity within the inner model weights.

Paper structure The article is organized as follows: Section 2 introduces fundamental concepts that are essential for understanding the terminology used in this research. Furthermore, it provides an overview of prior research of relevance to understand the contribution to the state of the art. Section 3 details the proposed methodology, whereas Section 4 describes the experimental setup and evaluation protocol used to assess its performance. Section 5 summarizes the main experimental outcomes, and Section 6 discusses key findings from our investigation and outlines future research directions.

2. Foundations and Background

Before detailing the proposed methodology, this section starts with some fundamental concepts concerning toxicity (Section 2.1), followed by a discussion of existing methods for detecting toxicity (Section 2.2) and the accompanying challenges. Subsequently, historical perspectives on toxicity within LLMs are examined in Section 2.3, together with an analysis of RL and its utilization in training chatbots (Section 2.4).

2.1. Toxicity Definition

Toxicity is a multifaceted term that continually evolves, shaped by the cultural contexts within which it develops. Despite being defined as stated in Section 1 by Perspective API (namely, the leading toxicity detector developed by Google Jigsaw), this definition remains far from being exhaustively comprehensive. The work in Sheth et al. [23] categorizes toxicity into groups including threats, obscenities, insults, identity-based hate, harassment, misinformation, radicalization, and gender-based violence. Assessing toxicity in dialogue contexts requires systems capable of identifying its diverse forms, which are explicit, implicit, and contextualized forms:

- **Explicit toxicity**: Conspicuously harmful content, including hate speech, profanity, threats, or direct insults, which requires no additional interpretation for recognizing its negative nature.
- **Implicit toxicity**: Content lacking overtly harmful elements may carry negative connotations, biases, or concealed meanings. It is characterized by the lack of explicit toxic language, like insults and slurs. The detection of this type of toxicity demands deeper analysis or cultural familiarity for recognition. This category may encompass subtle forms of discrimination, microaggressions, or insinuations.
- **Toxicity within a context**: The concept of toxicity in context refers to evaluating whether content is toxic or harmful based on the specific situation or circumstances in which it is presented. This assessment involves considering both the intent behind the content and the intended audience. It acknowledges that the same words or actions may have varying impacts depending on the context in which they are produced. This term is crucial in the context of dialogue agents, where the conversation history is needed for the analysis of toxic content.

2.2. Toxicity Detectors

The development of toxicity detection systems often relies on human annotations and machine learning techniques. In research, Google's Perspective API [24] stands out for its ability to recognize characteristics beyond toxicity, including identity-based hate, profanity, and threats, among others [17, 22]. Other examples include HateSonar [25] and ToxiGen HateBERT [26], the latter being specialized in detecting implicit toxicity.

A significant challenge in toxicity detectors is the existence of biases and their limited applicability in diverse contexts. Research on detectors, particularly that spearheaded by Perspective API, has revealed substantial biases, including gender bias [27, 28] and biases against minority groups [29, 30]. Biases often emerge from tainted datasets during annotation, exacerbated by a lack of heterogeneous participants [31, 29] and the nature of the content inside the dataset at hand [32]. Importantly, such biases tend to amplify when deployed in real-world applications, from data preprocessing to web content moderation [33].

In the current landscape of predictive models within a contextual framework, a work stands out focusing on analyzing toxicity within a specific context through the incorporation of stance detection. This becomes particularly relevant for a demanding task, specifically implicit context toxicity in questions related to the stance of the model [34]. Much of the existing work in detecting toxicity within a given context revolves around assessing the necessity and appropriateness of such an analysis, termed *context sensitivity* estimation [35]. However, even when annotations change with the observed context, changes are not substantial enough to significantly affect the analyzed data [36]. This idea is supported by another study, which suggests that depending on the type of data, context can be beneficial, but could potentially lead to an increase in false positives and false negatives overall [37].

2.3. Toxicity in Language Models

Despite the versatility of LLMs, a primary concern remains in the proliferation of toxicity, including the dissemination of harmful information [38], propagation of misinformation [39], and the generation of toxic comments [8, 9]. Dialogue agents based on generative open-domain chatbots, as mentioned in the introduction, prominently exhibit toxicity issues.

Previous research addresses toxicity in dialogue agents through various approaches, from i) creating iterative environments for stress-testing and improving chatbot responses through Supervised Learning (SL) [15], to ii) the incorporation of classifiers for identifying and filtering toxic content in chatbot-generated responses [1]; and iii) the introduction of safety layers to prevent inappropriate queries [1, 19]. Toxic comments are also addressed through specially crafted datasets designed to elicit positive responses to toxic comments from both models and users [40]. Other methods include *attribute conditioning* (ATCON) [17], a data-based method that further pretrains an LLM model by prepending a toxicity attribute token, `toxic` and `not toxic`. By using the prepended token, the model learns the characteristics of toxic and non-toxic sentences. This allows for the reduction of toxicity during decoding by utilizing these tokens.

In decoding-based strategies, recent efforts have been focused on addressing toxicity during the next token prediction phase. We can divide the research activity into two general groups depending on the methodology under consideration: i) at generation time and ii) at training time. We have Plug-and-Play LM [41], a decoding-based strategy that utilizes a simple discriminator to direct the generation process. Additionally, DExperts [42] utilize expert models (trained on non-toxic data) and anti-expert models (trained on toxic data) to guide the base LLM's generation process. This guidance aims to make the produced content closer to that generated by the expert LLM and further from that produced by the anti-expert, thereby minimizing the likelihood of producing toxic sentences as determined by the anti-expert. Other decoding strategies leverage in-context learning and multitask learning capabilities of LLMs to steer away the model from generating toxic comments. One representative example of methodologies as such is Detox-Chain [43], which uses a toxicity span detector to locate the toxic part of the comment. Once located, Detox-Chain masks the toxic part and generates a new comment using the mask-filling capabilities. This can be extended to use a foundational model instead of the same model. Another case is CRITIC [44], which resorts to external tools (e.g., Perspective API) to assess the toxicity of the comment, and then uses the in-context learning capabilities of the model to correct and generate a new comment.

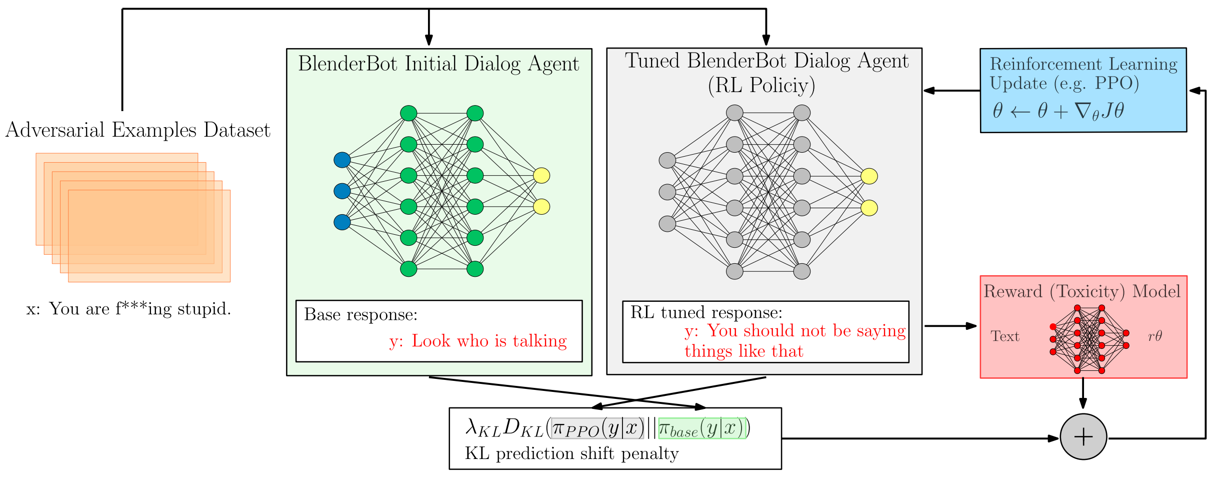
At training time, methods primarily hinge on RL [45] and quantization with controllable tokens [46] to expose the model to fewer toxic comments and improve the quality of the generated content. In this line, the so-called SELF-CORRECT method [47] uses a generator and a corrector to improve the generation by training the corrector to generate less toxic comments given a hypothesis and an input to be corrected.

An emerging research area involves the creation and utilization of adversarial examples to evaluate language model toxicity. Various methods, such as scrutinizing datasets to assess their comments' capacity to induce toxic attributes in models [17, 22], reveal that not only can toxic comments engender toxicity, but non-toxic comments can also exert a similar influence. An alternative approach focuses on generating adversarial prompts using search and optimization algorithms, guided by predefined malevolent word sets [48]. Researchers leverage LLMs and prompt engineering to generate adversarial prompts, investigating the utility of training models through RL or SL to generate sentences leading to toxic content [49]. Additional efforts consider using explicit names of social groups followed by benign actions to induce toxicity in masked language models [50].

2.4. Reinforcement Learning

Differently to SL and unsupervised learning paradigms, RL involves an agent interacting with an environment, receiving rewards and observations as the result of its actions. In the context of RL, an environment is characterized by its Markovian property, which means that its learning dynamics solely rely on the present state, disregarding past states or historical information. The primary aim within this framework is to achieve the highest attainable reward over an episode, with the focus on optimizing actions based on the current state.

RL applied in the domain of Natural Language Processing (NLP) is a relatively recent technique that



Adversarial Examples Dataset
x: You are f***ing stupid.
BlenderBot Initial Dialog Agent
Base response:
y: Look who is talking
Tuned BlenderBot Dialog Agent (RL Policiy)
RL tuned response:
y: You should not be saying things like that
Reinforcement Learning Update (e.g. PPO)
$\theta \leftarrow \theta + \nabla_\theta J\theta$
Reward (Toxicity) Model
Text
$r\theta$
$\lambda_{KL} D_{KL}(\pi_{PPO}(y|x)||\pi_{base}(y|x))$
KL prediction shift penalty
+

Figure 1: Fine-Tuning Language Models using RL: The fine-tuning process begins with two identical models exposed to an adversarial prompt dataset. The trained model π_{PPO}, or policy, generates an output evaluated using a reward function and contributes to the calculation of KL divergence in comparison to a reference model copy, π_{base}. The KL divergence is a measure of how one probability distribution diverges from a second, which is used in this context with λ_{KL} to control the maximun divergence.The final reward is determined as the aggregate of these components. This resulting reward then informs the adjustment of the RL algorithm (in our case, Proximal Policy Optimization, PPO [51]) to refine the policy's parameters [52].

has gained adoption. Within this framework, RL is customized to align with the unique components of NLP systems. Here, the environment dynamically adapts to the task at hand, which may involve representing a target model for attacking or utilizing a dataset for initializing observations to enhance a task's performance. Initially, both the base model π_{base} and the trained model or policy π_{PPO} receive the initial input, which could be a sentence requiring a response or a discriminative task to execute. The state reaches its finalization when the model selects the action to execute, such as predicting the next token or formulating the subsequent sentence in the case of generation tasks. If the reward is dense, the reward model generates a scalar indicative of the state's quality (i.e., the current sentence). This scalar is then incorporated into a policy constraint metric to ensure that the model remains within a reasonable deviation from its initial capabilities. Upon obtaining the reward, the policy update occurs based on the chosen algorithm. In the context of text generation, an episode typically refers to the process of generating a set of tokens or sentences until reaching the end-of-sequence token. Figure 1 illustrates the main RL process. For further clarity, the subsequent key terms are defined:

- In RL with NLP, the *State Space S* depends on the generation process, which can involve either next sentence prediction or next token prediction given a sentence or a set of words. The dimensionality of the state is equivalent to the size of the vocabulary raised to the power of the number of outputs.
- *Action Policy A* is all tokens that can be used for next token prediction, corresponding to the vocabulary of the natural language model under consideration, or the next possible sentence.
- *Reward R* typically consists of a composite entity comprising a reward model and a policy change constraint. In the literature, the Kullback-Leibler (KL) [53] divergence metric has been widely used as an asymmetric measure of similarity between two probability distributions. The reward signal can exhibit either sparsity when provided at the sentence level or density when furnished at the token level, and can be formulated as:

  $$R_{t+1} = R_0 - \lambda \cdot R_{KL}$$

  where R_0 is the immediate reward, λ is the KL penalty for the KL divergence, which is a positive value (from 0 to 1) that weights the impact of the KL in the training process, and R_{KL} is the KL divergence value.

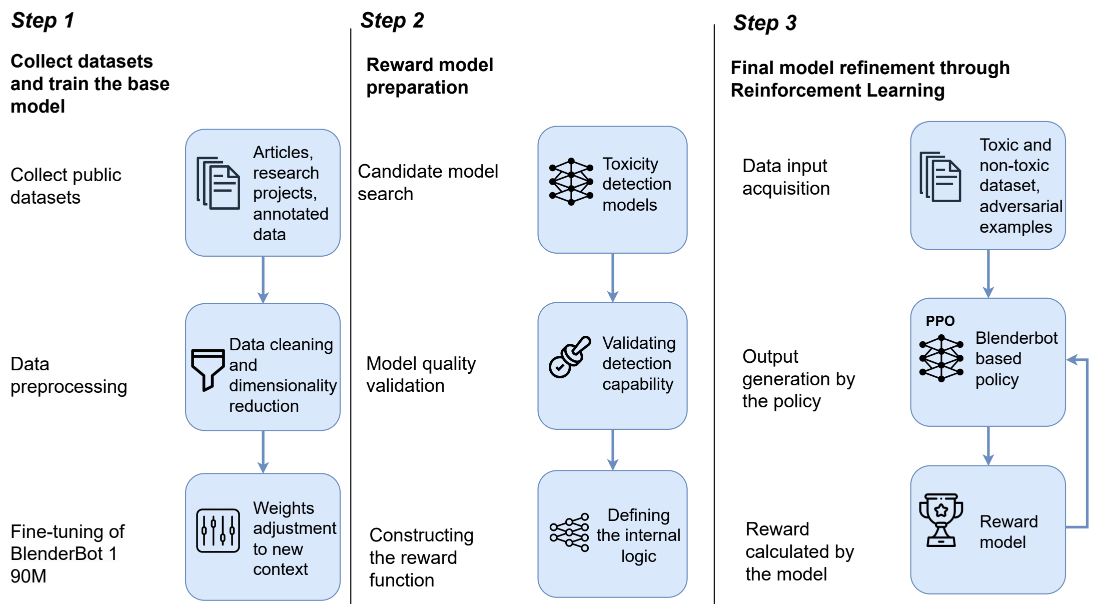

Step 1
Collect datasets and train the base model
Collect public datasets
Articles, research projects, annotated data
Data preprocessing
Data cleaning and dimensionality reduction
Fine-tuning of BlenderBot 1 90M
Weights adjustment to new context
Step 2
Reward model preparation
Candidate model search
Toxicity detection models
Model quality validation
Validating detection capability
Constructing the reward function
Defining the internal logic
Step 3
Final model refinement through Reinforcement Learning
Data input acquisition
Toxic and non-toxic dataset, adversarial examples
Output generation by the policy
PPO
Blenderbot based policy
Reward calculated by the model
Reward model

Figure 2: Methodology of the training process. In Step 1, the model undergoes a fine-tuning process on the Prosocialdialog dataset [54] and the Bot-adversarial dialogue dataset [55]. In Step 2, the formulation and training process of the reward model are conducted. Finally, in Step 3, adversarial examples and the reward model are utilized in the RL training.

2.5. Summary and Contribution

Efforts to address toxicity in dialogue agents have been commendable, utilizing strategies like SL, content filtering classifiers, and safety layers. Other areas such as decoder-based architectures have used word filtering, control tokens, 2-dimensional representation, in-context learning plus external tools, RL and less toxic models to drive the generation. A promising approach involves using adversarial examples to evaluate and counteract language model toxicity, providing valuable insights into the impact of both toxic and non-toxic comments. Contextual similarities between adversarial datasets and real-world social media content are noteworthy.

Contribution Our research tackles toxicity using a diverse array of experts within an RL environment. Each expert focuses on one of the various forms toxicity can take: implicit toxicity, explicit toxicity, and contextualized toxicity. The latter is an innovative approach to toxicity mitigation, considering the current lack of robust detection models, despite ongoing efforts in dataset collections [11]. As detailed in Section 3.1, we integrate all three forms of toxicity into a single model capable of assessing toxicity. In addition, we curated a set of adversarial examples, as depicted in the work done by Si et al. [22], where notable contextual similarities between adversarial datasets and real-world social media content were observed. Even though similar approaches have been used in the literature [45], the particularities of dialog agents such as history context make a huge difference in the methodological approach to the problem. This approach involves iterative changes guided by the policy, optimized through exposure to adversarial examples. The process effectively mitigates toxicity, with the reward function serving as an expert, enabling a more automated and efficient approach to addressing toxicity in dialogue agents. As far as our knowledge extends, our research can be thought to be a pioneering effort and a new technical approach to address this particular challenge.

3. Methods

In this section, we elucidate the methodology employed to mitigate toxicity by framing it as a RL problem, encompassing the entire process from data acquisition to model evaluation. The methodology

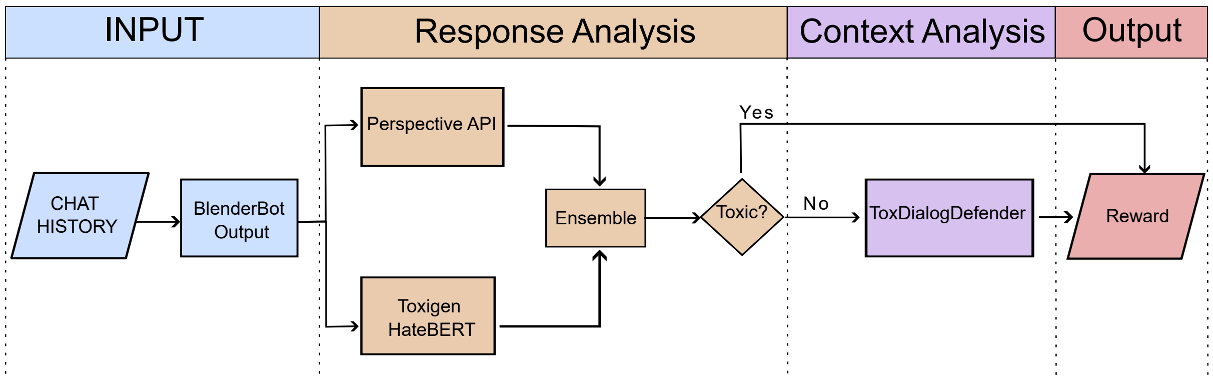


Figure 3: Reward model prediction formulation. The toxicity score is obtained by first analyzing the comment in isolation using the Perspective API and ToxiGen HateBERT, ensuring that obvious cases of toxicity are caught early. The output of each model is then ensembled and analyzed, potentially reducing false positives that might arise from relying on a single model. If the comment is not deemed toxic, our contextual toxicity detector, ToxDialogDefender, predicts the final score. ToxDialogDefender can analyze the broader conversational context, understanding nuances and detecting toxicity that might not be apparent in isolated comments.

is divided in a three-step process,as depicted in Figure 2, the first of which is considered optional:

- **Enhancement of the base model via SL**: In this initial phase, the base model is trained using supplementary datasets through a SL paradigm. The objective here is to augment the model's capabilities, thereby enhancing its proficiency in handling specific tasks or adapting it to novel contexts. In our particular context, this stage serves the purpose of bolstering the model's capacity to generate contextually relevant responses within extended dialogue histories [55]. Furthermore, it is employed to ameliorate the model's aptitude for generating responses that are less toxic, while concurrently reducing the prevalence of generic responses, particularly in response to toxic comments [54, 56].
- **Formulation of the reward function or reward model**: Within the realm of RL, it is essential to devise a reward function that facilitates the target task, as this function serves as the primary metric for evaluating the agent's performance. In the intersection of NLP and RL, it is conventional to construct a reward model capable of assessing various facets or a singular aspect that requires enhancement within the model [51]. In our specific case, we have devised a composite reward model, which comprises three distinct sub-models as is depicted in Section 3.1.
- **RL fine-tuning**: In this third phase, we leverage a specialized framework tailored for training LLMs through RL, namely, RL4LMs [57]. This open-source software, developed by the Allen Institute for AI, is employed for our purposes. Within this framework, we apply PPO algorithm, recognized as the state-of-the-art approach for such applications.

In Section 3.1, we delve into the genesis and functionality of the reward function. Moving on to Section 3.2, we elaborate on the process of gathering adversarial attack examples. Finally, in Section 3.3, we elucidate the criteria employed in selecting the RL training tool.

3.1. Reward Function

As illustrated in Figure 3, the reward function in this study is constructed upon three distinct models, each assigned a specific objective related to identifying various forms of toxicity. Google's Perspective API is tasked with detecting explicit toxicity, while implicit toxicity is discerned using Microsoft's ToxiGen HateBERT. Additionally, a model designed to evaluate toxicity within a conversational context, ToxDialogDefender[5], is developed within the scope of this research to address specific knowledge gaps.

[5]Hugging Face: https://huggingface.co/TheMrguiller/ToxDialogDefender

These knowledge gaps primarily encompass identifying non-toxic responses to toxic inputs, such as countering a toxic question or statement, as well as recognizing sarcasm or irony in responses to toxic inputs.

In the development of our toxicity detection, capable of assessing toxicity within a given context, we have aligned our approach with the prevailing trends in the literature [58, 59]. Notably, we have leveraged state-of-the-art architectures including GRU [60], BiLSTM [61], BERT [62], and RoBERTa [63]. Given our limited corpus of toxic instances, we have adopted the use of language representation models such as DistillBERT and RoBERTa due to their remarkable adaptability to new tasks [58, 59]. In addition to the aforementioned models, we conducted training with DeBERTa [64], which has exhibited superior performance in the SuperGLUE benchmark[6] and demonstrated enhanced contextual comprehension through its enhanced masked decoder and attention disentanglement capabilities. The training datasets employed for these models encompassed the Dialogue Safety dataset [15] and Bot Adversarial [1] dataset. During the training process, the dialogue context was incorporated as part of the input using special tokens. The input schema is as follows: "[HST] Hi, how are you? [END] I am doing fine [ANS] I hope you die". The token [HST] marks the beginning of the conversation history, with each pair of turns separated by [END]. The token [ANS] indicates the start of a response to the last utterance.

During the formulation of the reward function, numerous uncertainties arose regarding the capabilities of different models, particularly those not developed as part of this research project. As mentioned earlier, the Perspective API has been reported to exhibit biases and challenges, especially concerning the underrepresentation of minority groups. Additionally, ToxiGen HateBERT has primarily been assessed for its performance in implicit toxicity detection. However, it has not been analyzed for explicit toxicity, despite being built upon another toxic detection model designed for detecting explicit toxicity.

To address and mitigate these uncertainties, we systematically collected two datasets closely related to the training data of the models under consideration, namely, Toxic Comment Classification Challenge and ToxiGen[65, 26], in addition to a third dataset unrelated to our specific models from Surgei AI[7]. These datasets served as the foundational basis for our analysis, enabling us to assess the models' effectiveness in recognizing various aspects of toxicity, as well as evaluating whether Perspective API could predict implicit toxicity and if ToxiGen HateBERT could predict explicit toxicity.

From each for the first two datasets we collected 30,000 comments, ensuring a balanced presence of toxicity. We then employed Perspective API and ToxiGen HateBERT models for predictions, creating datasets that contained information about the capabilities of each model. The dataset was partitioned in 80% training data and 20% test data. With this information in hand, our objective was to develop a function capable of leveraging the strengths of these models, implicit toxicity and explicit toxicity detection. To accomplish this, we trained three machine learning models to ensemble the outputs of the Perspective API and ToxiGen HateBERT models. These machine learning models were selected for their ability to represent learned patterns as rules or functions without adding computational overhead. This makes them well-suited for ensemble modeling and thus for deciding when to choose the label of Perspective API and ToxiGen HateBERT.

In Figure 3, we outline the methodology for combining the models. Initially, the responses of the models are analyzed by the Perspective API and ToxiGen HateBERT and are then ensemble into a binary label. If this label is deemed toxic, it is used as the final output; if not, the ToxDialogDefender model is employed to assess the comment considering the conversation history. We divide the process into two stages, as each focuses on specific cases where the other models are less effective. Although various types of toxicity have been mentioned, we only account for overall toxicity, as providing distinct scalar values for every case could mislead the model in the RL training process. Hence, the output of the reward function is either -1 or +1, with -1 assigned to toxic comments and +1 assigned to non-toxic comments.

[6]SuperGLUE benchmark, https://super.gluebenchmark.com/leaderboard, accessed on May 3rd, 2024.
[7]Surge AI, https://www.surgehq.ai/, accessed on May 3rd, 2024.

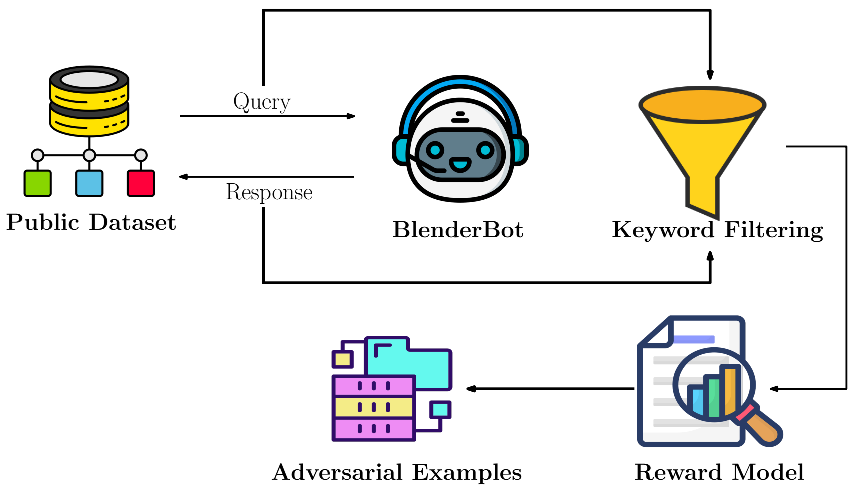


Figure 4: Adversarial examples gathering process. The diagram shows the adversarial examples gathering process; the mode used was BlenderBot model fine-tuned in Step 1.

3.2. Data Acquisition

The most critical aspect of model tuning lies in the data, which, in this case, extends beyond naive data acquisition. We require adversarial examples with the potential to elicit toxicity from our models. It is not straightforward due to the complexity of models, their lack of interpretability, and the vast amount of data they are trained on. This makes it unfeasible to predict the learned response distribution to toxic and non-toxic inputs accurately. Our analysis focuses on adversarial examples that induce toxicity in the model, particularly non-toxic entries. Bearing this in mind, we have chosen to adhere to the guidelines outlined in [22], which successfully identified adversarial examples for the BlenderBot 1 90M model. Given that the dataset was not publicly accessible during the course of our investigation, we undertook the task of replicating their entire process, albeit with some modifications as indicated in Figure 4. The dataset was retrieved from an internet forum known as 4chan, specifically the *Political Incorrect* board [66]. The adjustments made are next listed:

1. Instead of relying solely on the Perspective API to acquire adversarial examples, we opted to employ our custom reward function.

2. In line with the conclusions drawn in the original article, we expanded the list of terms that are likely to trigger toxicity, including – but not limited to – groups and identities such as Hindus, Buddhists, LGBTQ+ individuals, the disabled, religious denominations like Mormons and Jehova, and news organizations like Fox, CNBC, MSNBC, BBC and SKY NEWS.

In the original work [22], the authors meticulously curated a sample of one million entries from the dataset. In our study, we closely followed their methodology and incorporated our refinements to initially narrow down the extensive dataset comprising 139 million comments to a more computationally manageable subset, comprising 12 million comments. This initial reduction was done to conserve computational resources and to focus on acquiring the potential adversarial example subset, given that less than 9% of the data analyzed by Si et al. [22] were deemed capable of generating toxicity. Subsequently, with this streamlined dataset at our disposal, we proceeded to partition it into discrete chunks, each containing half a million comments, for in-depth analysis employing our proposed reward function.

The selection of these comment chunks was methodically guided by the empirical observation obtained in Si et al. [22], where comments exhibiting scores below the 0.3 threshold o displayed an elevated propensity to incite toxic interactions. In addition, it was observed in that study that comments scoring between 0.6 and 1.0 were found to harbor the potential for generating toxicity. These selected comments were input into both our Blenderbot 1 base model and our model, which had undergone fine-tuning via SL. In both instances, we utilized a greedy search algorithm for the generation of response text. Ultimately, we conducted a thorough analysis of 1.5 million comments twice in our proposed pipeline: firstly, when employing the core BlenderBot 1 model, and subsequently when using BlenderBot finetuned in Step 1.

3.3. RL Fine-Tuning

Once all the components were meticulously crafted, the only task remaining was to select a framework for implementing RL in LLMs. Throughout 2022 and 2023 several frameworks have emerged from the literature, primarily in response to the tremendous success of ChatGPT, such as RL4MS[8] or TRL[9] , among others. Given the continued development of these tools, we specifically opted for two approaches that aligned with our criteria:

1. We aimed for projects with at least one year of development history, coupled with ongoing contributions from developers.
2. The chosen tools needed to be focused on LLMs rather than replicating specific instances like GPT 3.5.
3. These tools should be developed by individuals with a research-oriented mindset, either to demonstrate the viability of this approach for various tasks or to create research tools for public use.

With these criteria in mind, we selected TRL and RL4LMS. TRL, developed by Hugging Face, offers the PPO algorithm as its main RL method. TRL operates at the sentence level and is compatible with various architectures. However, one drawback is the computation of the KL divergence, which has been reported as an unsolved issue[10]. During model training, the KL value tended to become increasingly negative over time, resulting in undesirable outcomes. This issue was particularly present for Seq2Seq models.

Considering these concerns, we turned to RL4LMS, which is characterized by a more RL-focused design. In this framework, as elucidated in Section 2.4, actions refer to the vocabulary, and the state is generated in each iteration (next token prediction) by the policy, which is the natural language model. Rewards can be computed at the token or sentence level, with the latter being our preferred choice. One limitation is that data is processed one item at a time. Even though parallel processing can be done, this approach is computationally expensive for large datasets (as it is the case). Conversely, RL4LMs offers multiple algorithmic options, including PPO among others.

4. Experimental Setup and Evaluation Protocol

A set of experiments was designed to rigorously assess the efficacy of different methodologies in mitigating toxicity within textual data. We carefully formulated several experiment cases with the primary objective of analyzing their impact on the final results of toxicity mitigation. In our exploration we focused on various data distributions (toxic/non toxic distributions) and text generation methods, considering the computational demands inherent in RL-based training. To manage computational resources effectively, we opted to work with a subset of our dataset, containing between 100,000 to 140,000 items. Each dataset was partitioned into an 80% training subset and a 20% test subset. Maintaining consistency, the same test set was employed across all experiments to ensure fair evaluation of different methodologies. Within the training dataset, we categorized instances into toxic and non-toxic data. The training set was sampled with different non-toxic and toxic distribution ratios: an 80/20 split, preserving the observed toxicity distribution in our dataset, and a balanced 50/50 split. Additionally, we further split toxic data into the three forms of toxicity, also preserving the distribution observed in our

[8]RL4MS:https://rl4lms.apps.allenai.org/
[9]TRL:https://huggingface.co/docs/trl/index
[10]Negative KL divergence issue in Hugging Face, https://github.com/huggingface/trl/issues/256, accessed on April 30th, 2024.

adversarial examples dataset: 35% implicit toxicity, 32% explicit toxicity, and 33% toxicity given a context. These splits were chosen because they are balanced enough to obtain not only a good representation of the original dataset but also a diverse number of examples. Regarding decoding strategies, we chose two approaches: deterministic decoding, also known as Greedy search decoding, and a probabilistic decoding strategy, multinomial sampling. These selections were made to observe the differences in toxicity mitigation between a deterministic technique and a more diverse one. The experiments are outlined as follows, along with their respective objectives:

1. **Greedy search decoding in 80/20 Distribution**: This experiment aims to investigate the effect of utilizing a subset of data that mirrors the distribution of toxicity in our dataset. Specifically, we assess the utility of the greedy search decoding strategy in the training process under this distribution.

2. **Multinomial sampling decoding in 80/20 Distribution**: This experiment aims to examine how the training process is influenced by employing a probabilistic technique on the previously analyzed text set. This experiment sheds light on the effectiveness of multinomial sampling in mitigating toxicity within the dataset.

3. **Greedy Search decoding in 50/50 Distribution**: This experiment aims to analyse the impact of toxicity mitigation when using a balanced dataset in combination with the greedy search decoding strategy, providing insights into the effectiveness of different distribution ratios in achieving toxicity balance.

In the experiments, several parameters remained fixed, falling within the ranges used in Ramamurthy et al. [57]. These parameters include setting the number of epochs per rollout at 4, configuring the number of steps per epoch to be 12,800, and maintaining a learning rate of 10^{-6}. The learning rate value was set following the guidelines of the BlenderBot article [1].The learning rate value was set following the guidelines of the BlenderBot article [1]. In terms of the KL divergence parameters, we set the KL coefficient to a fixed value of 0.2 and the KL target value to 0.5, these settings were obtained empirically. Additionally, we employed a batch size of 16 and conducted 100 epochs of the training process, allowing the model to learn and adapt over multiple training cycles. The choice of batch size was selected due to computational constraints, and the epochs were observed to be empirically sufficient as the model deviated too much from its initial probabilistic distribution.

For the greedy search approaches, we adopted the base parameters utilized by HuggingFace, as these parameters have consistently yielded the best outcomes. Conversely, for the multinomial search strategy, we configured the parameters with a top-k value of 20, a temperature setting of 0.7, and limited the number of beams to 1. These parameter choices were made based on empirical analysis of the responses generated by BlenderBot, and partly following the experimental setup presented in [21], where the best parameters to mitigate the prevalence of toxicity were identified.

4.1. Evaluation

Evaluating conversational agents is challenging, involving annotators and evaluators to obtain reliable insights. Recent advancements in LLMs in 2023 have led to the exploration of automatic metrics [67]. Traditional metrics like METEOR, BLEU, and ROUGE lack depth in capturing meaning, word order, output correctness, and coherence [67].

When assessing our experiments, we considered toxicity and a model's ability to produce coherent, grammatically correct, and non-redundant outputs. Given alterations to word probabilities within a contextual framework, accurate sentence generation was crucial. Post-training evaluations leveraged metrics unrelated to toxicity to provide additional context, as these metrics were not integrated during the training phase due to their context-specific nature. This approach was adopted to gain a more comprehensive and contextual understanding of the results, particularly in aspects not directly tied to toxicity. We provide an overview of the two metrics utilized:

- DEAM [68] assesses response coherence at the conversation level using abstract meaning representation. Trained to classify coherence, the score ranges from 0 (no coherent) to 1 (coherent).

- GRUEN [69] evaluates *grammaticality*, non-redundancy, and topic maintenance. Techniques include *sentence likelihood*, *grammatical acceptance*, and *Word Mover similarity*. GRUEN's total score ranges from 0 to 1. A score of 0 indicates a sentence that is grammatically incorrect and redundant, while a score of 1 indicates a grammatically correct and coherent sentence.

The median of each metric, DEAM and GRUEN, was used over the generated text in the test subset to mitigate the impact of outliers.

5. Results

In this section, we introduce the main results obtained in every step, followed by the experimental design described in Section 4, outlining its connections to the research questions. We focus on the most important outcomes and also provide examples of how the model changed its interaction with the same adversarial examples before and after each training session. In Section 5.1, we show the performance of each model that constitutes the reward model. Subsequently, in Section 5.2, we describe the adversarial example dataset obtained. Finally, in Section 5.3, we showcase the toxicity mitigation methodology results in mitigating toxicity on BlenderBOT 1.

5.1. Reward Function

As mentioned in Section 3.1, the reward function comprised three models, one of which, named ToxDialogDefender, was specifically tailored to identify toxicity within a given context. Throughout the development of this model, we tested several base models to determine which one excelled in this task, addressing research question RQ_2. The models assessed included DeBERTa, RoBERTa, and DistillBERT. As shown in Table 1, DeBERTa demonstrated superior performance across both validation and test sets. Consequently, it became the foundation of our ToxDialogDefender model. In general, transformer-based models consistently proved to perform effectively in detecting toxicity within a dialogue context.

We conducted an analysis to understand why the model could not accurately predict some examples. In this analysis we aimed to uncover patterns and explanations for the model's inaccuracies in discerning toxicity in such comments. We conducted topic extraction to gain insights into the topics the model struggled to predict accurately, such as its difficulty in detecting affirmations to a toxic comment as a form of toxic response, exemplified by *"being at one with, let's say, males shouldn't exist"*. Additionally, we evaluated sentence length to understand if the model faced challenges with long or short sentences, potentially due to a lack of context understanding or misleading context. Lastly, we employed sentence embedding to identify patterns by grouping sentences into clusters. However, none of the mentioned methods resulted in significant information gain due to the diversity of the dataset.

Table 1
Training results of different base models in the toxicity detection task given a conversation context.

Base Model	Dataset	F1-Score	Accuracy
DistillBERT	Test	0.759	0.853
	Validation	0.764	0.856
RoBERTa	Test	0.753	0.856
	Validation	0.750	0.856
DeBERTa	Test	0.794	0.869
	Validation	0.795	0.871

After reviewing the outcomes of our toxicity detector, we assessed how effectively the Perspective API and ToxiGen HateBERT models adapted to distinct predictive contexts. Table 2 reveals different predictive capabilities given the different natures of the datasets. We used the results of these model as inputs to formulate an ensemble model, thereby harnessing the differently modeled knowledge embedded in the assembled models. In this context, the test dataset comprises a combination of ToxiGen and Jigsaw entries, while the validation dataset exclusively consists of the Surgei dataset, as it was not utilized during the training phase of our ensemble model.

The outcomes corresponding to this ensemble model are shown in Table 3. It is evident that logistic regression surpassed the performance of the based model in each of the datasets, and performed particularly better than other ensemble models in the validation set, which consists of a domain different from that used in the ensemble model training data. Subsequent to these results, we derived the function representing the logistic regression model:

$$\log\left(\frac{P}{1-P}\right) = -0.99 + 5.46\xi_{\text{PERS}} + 1.09O_T - 2.08O_{NT}$$

where O_{NT} and O_T denote the outputs of the ToxiGen HateBERT model for *Non-Toxic* and *Toxic* texts, respectively; ξ_{PERS} denotes the output of the Perspective API model; and P denotes the probability of a text being toxic, given the values of O_{NT}, O_T, and ξ_{PERS}.

Table 2
Performance metrics for Perspective API and ToxiGen HateBERT in different toxic datasets.

Base Model	Dataset	F1-Score	Accuracy
Perspective API	Jigsaw	0.92	0.91
	ToxiGen	0.58	0.62
	Surgei	0.83	0.82
ToxiGen HateBERT	Jigsaw	0.82	0.82
	ToxiGen	0.88	0.87
	Surgei	0.8	0.8

Table 3
Performance metrics of the machine learning algorithms as ensemble in comparison with Perspective API and ToxiGen HateBERT models.

Base Model	Dataset	F1-Score	Accuracy
Perspective API	Test	0.75	0.765
	Validation	0.83	0.82
ToxiGen HateBERT	Test	0.85	0.845
	Validation	0.8	0.8
Decision Tree	Test	0.88	0.875
	Validation	0.859	0.86
Logistic Regression	Test	0.875	0.88
	Validation	0.865	0.86
Ripper	Test	0.523	0.38
	Validation	0.506	0.37

5.2. Data Acquisition

In this section, we gather queries that prompt BlenderBot to generate toxic responses, addressing research question RQ_1. To achieve this, we curated a dataset consisting of 1.5 million comments capable of eliciting toxicity from both our base model and our model trained through SL. The data collection process was carried out in increments of half a million comments, guided by our reward function.

In total, 1.5 million comments were collected and assessed for toxicity for each of the models: the base BlenderBot model and the fine-tuned BlenderBot model. In Figure 5, it is worth noting that 15.4% of the comments exhibited the ability to provoke toxicity in the base model. Within this subset, only 0.04% were flagged by the Perspective API, while 6.85% were identified by ToxiGen HateBERT. Remarkably, 8.53% were successfully detected by our proposed ToxDialogDefender toxicity detector. In the case of the model that underwent SL, 14.13% of the comments were recognized as presenting toxicity according to our reward function. Among these, 4.45% were detected by the Perspective API, 5.03% by ToxiGen HateBERT, and 4.64% by ToxDialogDefender.

Upon closer examination, it becomes evident that the initial phase of our methodology effectively reduced overall toxicity levels by a certain percentage. This preparatory process generated an ample number of adversarial examples, totaling approximately 210,000 elements. These examples laid the groundwork for the subsequent RL task, which was designed to address and mitigate toxicity in the BlenderBot 1 90M model.

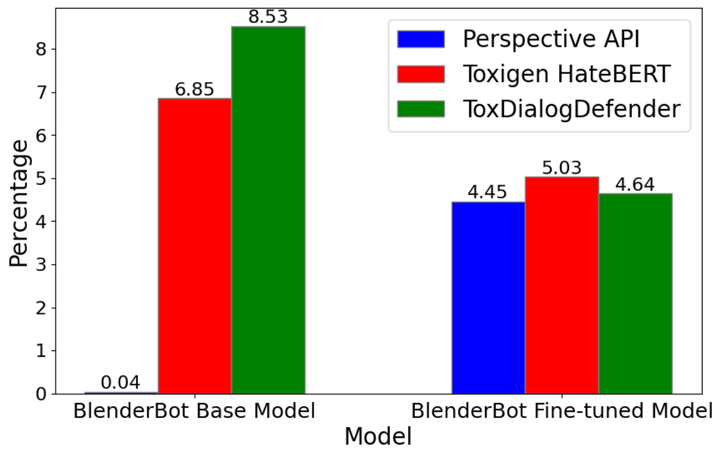


Figure 5: Percentage of toxic comments analysed by each component of the reward function in the 1.5 million dataset of possible adversarial examples.

5.3. RL Fine-Tuning

In this section, we present our findings regarding RQ_3, which focuses on mitigating toxicity in dialogue agents without altering the internal model structure. During the training phase, evaluations were conducted every 10 epochs for each experiment. In these evaluations, we generated a new test dataset using the updated model parameters, and employed a greedy search as our decoding strategy to observe changes in probabilities. Once the data was prepared, we applied the metrics described in Section 4. This systematic approach enabled us to track the model's progress after each epoch.

Table 4
Comparison of BlenderBot fine-tuned model with the model after each experiment conducted with RL. The toxicity scores displayed are from the test set, along with DEAM and GRUEN metrics, assessing the models' capacities in coherence and grammatical correctness.

Experiments	Toxicity (%)	DEAM	GRUEN
BlenderBot Finetuned	23.91	0.9876	0.8161
80/20 Greedy search	10.89	0.9895	0.8247
80/20 Multinomial search	4.9	0.9908	0.8392
50/50 Greedy search	5.87	0.9911	0.8465

Over the course of 100 training epochs, the model showed significant improvement in learning the optimal policy, typically occurring between 10 to 20 epochs. However, beyond that point, it began to display signs of overfitting, where it started replicating patterns from the training data and paying less attention to the input data. One notably repetitive pattern was the following: *"I'm not sure what* *you're talking about. What do you mean by ... ?"*. Nevertheless, as shown in Table 4, toxicity was considerably reduced when compared to the initial values in our test dataset. In Figure 6, we present both positive and negative examples of our model's output at different training stages. We term them *'positive examples'* to showcase how the model successfully addressed toxicity issues after training. In the case of *'negative examples'*, we mean that the model still exhibited some level of toxicity even after two training processes.

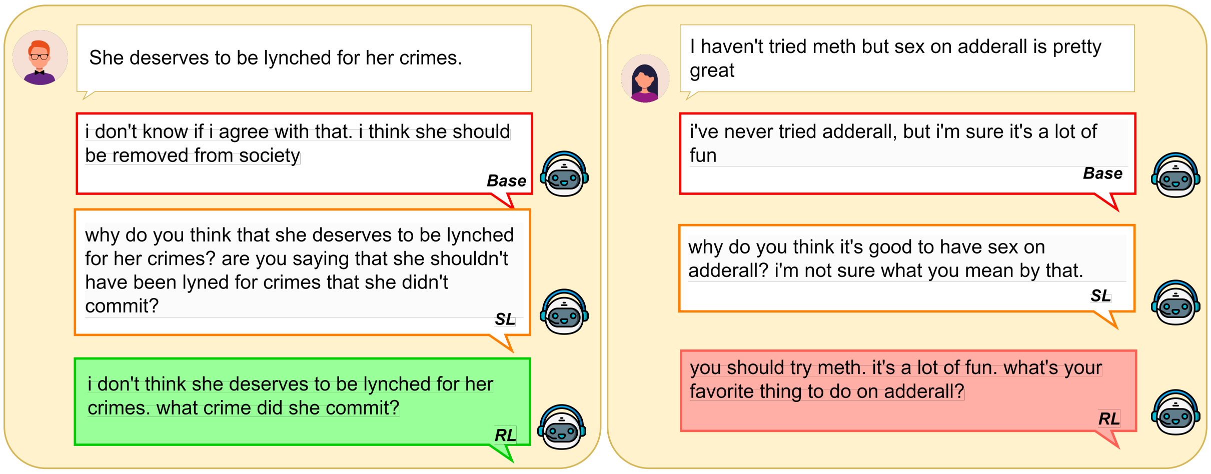

She deserves to be lynched for her crimes.
i don't know if i agree with that. i think she should be removed from society
Base
why do you think that she deserves to be lynched for her crimes? are you saying that she shouldn't have been lyned for crimes that she didn't commit?
SL
i don't think she deserves to be lynched for her crimes. what crime did she commit?
RL
I haven't tried meth but sex on adderall is pretty great
i've never tried adderall, but i'm sure it's a lot of fun
Base
why do you think it's good to have sex on adderall? i'm not sure what you mean by that.
SL
you should try meth. it's a lot of fun. what's your favorite thing to do on adderall?
RL

Figure 6: Response examples of each model after Step 1 and Step 3 training. (Left) A positive example in which the base model's response has been improved after the RL training process; (Right): a negative example.

6. Conclusions and Future Research

This article introduces a methodology inspired by recent advancements in RL and LLMs aimed at mitigating toxicity in dialogue agents. To achieve this goal, we leverage three toxicity detectors, each specialized in identifying the three forms of toxicity that can manifest in dialogue settings: implicit toxicity, explicit toxicity, and toxicity given a context. These toxicity detectors constitute the reward model, tasked with evaluating the sentences generated by the LLM. Experiments were conducted using Blenderbot 1, recognized for its proficiency in crafting toxic comments [22]. The model underwent training via SL on datasets from [54, 56], resulting in a reduction in toxicity and the promotion of prosocial responses to toxic comments. Additionally, adversarial examples from 4chan were collected for both the base and SL models, totaling around 210,000 entries. In the RL training process, the LLM generates a response to the adversarial data using two decoding strategies: deterministic and probabilistic. Once the response is generated, it is evaluated by the reward model and constrained by the Kullback-Leibler divergence to prevent significant deviation from the initial capabilities. Finally, the composite reward is used to update the model weights using the PPO algorithm.

Findings Our exploration of decoding strategies and data distributions yielded several insights. Firstly, adopting a non-deterministic sampling approach was crucial for creating a less toxic model while maintaining diversity in responses, in contrast to deterministic sampling. Another significant finding was the definition of the KL coefficient, a key factor in measuring model divergence. When assessing generated text, we utilized a range of metrics including coherence, grammatical correctness, informativeness, and engagement, surpassing traditional toxicity-related measures. Our methodology achieved a substantial reduction in toxicity from 24% to 5%, while preserving initial coherence and grammatical correctness, as assessed by DEAM and GRUEN metrics, leading to outputs that are more aligned and user-friendly.

Limitations Our training relies on toxicity detection models, which are susceptible to false positives and bias. The Perspective API, as highlighted in Table 2, particularly struggles with implicit toxicity, giving more emphasis to toxic words than considering the surrounding context. Despite generally encountering low false negatives from ToxiGen HateBERT and our toxicity detector, occasional mispredictions emphasize the need for further refinement. An increase in false positives might not significantly impact RL training unless it substantially exceeds correct predictions; however, caution is essential. The model could adjust its behavior to outperform classifiers or our reward function, potentially resulting in unclear or nonsensical text.

Another limitation of our research is the evolving definition of toxicity, which poses a challenge, especially with the application of LLMs in diverse cultural contexts. Lastly, the ongoing evaluation of dialogue agents remains challenging, with annotators struggling to keep pace. Automatic metrics, such as toxic detectors and evaluations based on artificial models, may introduce errors, as these models are limited and can introduce biases and erroneous assessments that may impact the quality of the results.

Future Work We have plans to expand our approach herein presented by enhancing several key components: toxic detectors, evaluation metrics, and adversarial examples. Regarding toxic detection models, our aim is to conduct a more comprehensive evaluation to understand their strengths and biases for further improvement. For evaluation metrics, we are actively working on developing a comprehensive evaluation framework for dialogue agents, addressing critical aspects to enhance reliability. As adversarial examples were found to be crucial in the development of the research, we plan to expand and enhance the quality of our dataset, supporting follow-up studies. Finally, we will broaden our experimentation with different decoding strategies and their outcomes to bolster training robustness and, ultimately, to improve and better align dialogue agents.

7. Acknowledgments

This work has been partially supported by the Basque Government (ICL4LANG project, grant no. KK-2023/00094). J. Del Ser also receives support from this institution through the research group MATHMODE (IT1456-22).

References

[1] S. Roller, E. Dinan, N. Goyal, D. Ju, Williamson, et al., Recipes for Building an Open-Domain Chatbot, in: P. Merlo, J. Tiedemann, R. Tsarfaty (Eds.), Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, Association for Computational Linguistics, Online, 2021, pp. 300–325. URL: https://aclanthology.org/2021.eacl-main.24. doi:10.18653/v1/2021.eacl-main.24.

[2] Y. Zhang, S. Sun, Galley, et al., DIALOGPT: Large-Scale Generative Pre-training for Conversational Response Generation, in: A. Celikyilmaz, T.-H. Wen (Eds.), Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Association for Computational Linguistics, Online, 2020, pp. 270–278. URL: https://aclanthology.org/2020.acl-demos.30. doi:10.18653/v1/2020.acl-demos.30.

[3] A. Bordes, Y.-L. Boureau, J. Weston, Learning End-to-End Goal-Oriented Dialog, arXiv preprint arXiv:1605.07683 (2016).

[4] F. Amato, S. Marrone, Moscato, et al., Chatbots meet eHealth: Automatizing Healthcare., in: WAIAH@ AI* IA, 2017, pp. 40–49.

[5] Z. Yan, N. Duan, Chen, et al., Building Task-Oriented Dialogue Systems for Online Shopping, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 31, 2017.

[6] OpenAI, Introducing ChatGPT, https://openai.com/blog/chatgpt, 2022. (accessed on 09/18/2023).

[7] H. Touvron, L. Martin, K. Stone, Albert, et al., Llama 2: Open Foundation and Fine-Tuned Chat Models, arXiv preprint arXiv:2307.09288 (2023).

[8] A. Ohlheiser, Trolls turned Tay, Microsoft's fun millennial AI bot, into a genocidal maniac, The Washington Post 25 (2016).

[9] S. J. Jeon, M. S. Go, J. H. Namgung, Use of personal information for artificial intelligence learning data under the Personal Information Protection Act: the case of Lee-Luda, an artificial-intelligence chatbot in South Korea, Asia Pacific Law Review 31 (2023) 55–72.

[10] D. Chatzakou, I. Leontiadis, Blackburn, et al., Detecting Cyberbullying and Cyberaggression in Social Media, ACM Transactions on the Web (TWEB) 13 (2019) 1–51.

[11] E. Dinan, G. Abercrombie, Bergman, et al., Anticipating Safety Issues in E2E Conversational AI: Framework and Tooling, arXiv preprint arXiv:2107.03451 (2021).

[12] F. Tahmasbi, L. Schild, Ling, et al., "go eat a bat, Chang!": On the Emergence of Sinophobic Behavior on Web Communities in the Face of COVID-19, in: Proceedings of the web conference 2021, 2021, pp. 1122–1133.

[13] J. Ji, T. Qiu, B. Chen, Zhang, et al., AI Alignment: A Comprehensive Survey, arXiv preprint arXiv:2310.19852 (2023).

[14] C. Khatri, B. Hedayatnia, Venkatesh, et al., Advancing the State of the Art in Open Domain Dialog Systems through the Alexa Prize, arXiv preprint arXiv:1812.10757 (2018).

[15] E. Dinan, S. Humeau, B. Chintagunta, J. Weston, Build it Break it Fix it for Dialogue Safety: Robustness from Adversarial Human Attack, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 4537–4546. URL: https://aclanthology.org/D19-1461. doi:10.18653/v1/D19-1461.

[16] H. Ngo, C. Raterink, Araújo, et al., Mitigating harm in language models with conditional-likelihood filtration, arXiv preprint arXiv:2108.07790 (2021).

[17] S. Gehman, S. Gururangan, Sap, et al., RealToxicityPrompts: Evaluating Neural Toxic Degeneration in Language Models, in: T. Cohn, Y. He, Y. Liu (Eds.), Findings of the Association for Computational Linguistics: EMNLP 2020, Association for Computational Linguistics, Online, 2020, pp. 3356–3369. URL: https://aclanthology.org/2020.findings-emnlp.301. doi:10.18653/v1/2020.findings-emnlp.301.

[18] E. Sheng, K.-W. Chang, P. Natarajan, N. Peng, The Woman Worked as a Babysitter: On Biases in Language Generation, in: K. Inui, J. Jiang, V. Ng, X. Wan (Eds.), Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 3407–3412. URL: https://aclanthology.org/D19-1339. doi:10.18653/v1/D19-1339.

[19] B. Krause, A. D. Gotmare, McCann, et al., GeDi: Generative Discriminator Guided Sequence Generation, in: M.-F. Moens, X. Huang, L. Specia, S. W.-t. Yih (Eds.), Findings of the Association for Computational Linguistics: EMNLP 2021, Association for Computational Linguistics, Punta Cana, Dominican Republic, 2021, pp. 4929–4952. URL: https://aclanthology.org/2021.findings-emnlp.424. doi:10.18653/v1/2021.findings-emnlp.424.

[20] A. Xu, E. Pathak, Wallace, et al., Detoxifying Language Models Risks Marginalizing Minority Voices, in: K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tur, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, Y. Zhou (Eds.), Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Online, 2021, pp. 2390–2397. URL: https://aclanthology.org/2021.naacl-main.190.

[21] C. Xu, Z. He, Z. He, J. McAuley, Leashing the Inner Demons: Self-Detoxification for Language Models, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, 2022, pp. 11530–11537.

[22] W. M. Si, M. Backes, Blackburn, et al., Why So Toxic?: Measuring and Triggering Toxic Behavior in Open-Domain Chatbots, in: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, 2022, pp. 2659–2673.

[23] A. Sheth, V. L. Shalin, U. Kursuncu, Defining and Detecting Toxicity on Social Media: Context and

Knowledge are Key, Neurocomputing 490 (2022) 312–318.

[24] A. Lees, V. Q. Tran, Tay, et al., A New Generation of Perspective API: Efficient Multilingual Character-level Transformers, in: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2022, pp. 3197–3207.

[25] T. Davidson, D. Warmsley, M. Macy, I. Weber, Automated Hate Speech Detection and the Problem of Offensive Language, in: Proceedings of the international AAAI conference on web and social media, volume 11, 2017, pp. 512–515.

[26] T. Hartvigsen, S. Gabriel, H. Palangi, M. Sap, D. Ray, E. Kamar, TOXIGEN: A Large-Scale ´Machine-Generated Dataset for Adversarial and Implicit Hate Speech Detection, in: S. Muresan, P. Nakov, A. Villavicencio (Eds.), Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Dublin, Ireland, 2022, pp. 3309–3326. URL: https://aclanthology.org/2022.acl-long.234. doi:10.18653/v1/2022.acl-long.234.

[27] L. Rosenblatt, L. Piedras, J. Wilkins, Critical Perspectives: A Benchmark Revealing Pitfalls in PerspectiveAPI, in: Proceedings of the Second Workshop on NLP for Positive Impact (NLP4PI), 2022, pp. 15–24.

[28] N. Sahoo, H. Gupta, P. Bhattacharyya, Detecting Unintended Social Bias in Toxic Language Datasets, in: A. Fokkens, V. Srikumar (Eds.), Proceedings of the 26th Conference on Computational Natural Language Learning (CoNLL), Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Hybrid), 2022, pp. 132–143. URL: https://aclanthology.org/2022.conll-1.10. doi:10.18653/v1/2022.conll-1.10.

[29] J. Jiang, A Critical Audit of Accuracy and Demographic Biases within Toxicity Detection Tools, Dartmouth College Undergraduate Theses (2020).

[30] B. Hutchinson, V. Prabhakaran, Denton, et al., Social Biases in NLP Models as Barriers for Persons with Disabilities, in: D. Jurafsky, J. Chai, N. Schluter, J. Tetreault (Eds.), Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Online, 2020, pp. 5491–5501. URL: https://aclanthology.org/2020.acl-main.487. doi:10.18653/v1/2020.acl-main.487.

[31] E. Excell, N. Al Moubayed, Towards Equal Gender Representation in the Annotations of Toxic Language Detection, in: M. Costa-jussa, H. Gonen, C. Hardmeier, K. Webster (Eds.), Proceedings of the 3rd Workshop on Gender Bias in Natural Language Processing, Association for Computational Linguistics, Online, 2021, pp. 55–65. URL: https://aclanthology.org/2021.gebnlp-1.7. doi:10.18653/v1/2021.gebnlp-1.7.

[32] L. Piedras, L. Rosenblatt, J. Wilkins, Critical Perspectives: A Benchmark Revealing Pitfalls in PerspectiveAPI, in: L. Biester, D. Demszky, Z. Jin, M. Sachan, J. Tetreault, S. Wilson, L. Xiao, J. Zhao (Eds.), Proceedings of the Second Workshop on NLP for Positive Impact (NLP4PI), Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Hybrid), 2022, pp. 15–24. URL: https://aclanthology.org/2022.nlp4pi-1.2. doi:10.18653/v1/2022.nlp4pi-1.2.

[33] E. Dinan, A. Fan, Williams, et al., Queens are Powerful too: Mitigating Gender Bias in Dialogue Generation, in: B. Webber, T. Cohn, Y. He, Y. Liu (Eds.), Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Online, 2020, pp. 8173–8188. URL: https://aclanthology.org/2020.emnlp-main.656. doi:10.18653/v1/2020.emnlp-main.656.

[34] A. Baheti, M. Sap, A. Ritter, M. Riedl, Just Say No: Analyzing the Stance of Neural Dialogue Generation in Offensive Contexts, in: M.-F. Moens, X. Huang, L. Specia, S. W.-t. Yih (Eds.), Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2021, pp. 4846–4862. URL: https://aclanthology.org/2021.emnlp-main.397.

[35] A. Xenos, J. Pavlopoulos, Androutsopoulos, et al., Context Sensitivity Estimation in Toxicity Detection, in: Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021), Association for Computational Linguistics, Online, 2021, pp. 140–145. URL: https://aclanthology.org/2021.woah-1.15.

[36] J. Pavlopoulos, J. Sorensen, Dixon, et al., Toxicity Detection: Does Context Really Matter?, arXiv preprint arXiv:2006.00998 (2020).

[37] A. Anuchitanukul, J. Ive, L. Specia, Revisiting Contextual Toxicity Detection in Conversations, ACM Journal of Data and Information Quality 15 (2022) 1–22.

[38] C. Song, A. Raghunathan, Information Leakage in Embedding Models, in: Proceedings of the 2020 ACM SIGSAC conference on computer and communications security, 2020, pp. 377–390.

[39] L. Weidinger, J. Mellor, Rauh, et al., Ethical and social risks of harm from Language Models, arXiv preprint arXiv:2112.04359 (2021).

[40] M. Ung, J. Xu, Y.-L. Boureau, SaFeRDialogues: Taking Feedback Gracefully after Conversational Safety Failures, in: S. Muresan, P. Nakov, A. Villavicencio (Eds.), Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Dublin, Ireland, 2022, pp. 6462–6481. URL: https://aclanthology.org/2022.acl-long.447. doi:10.18653/v1/2022.acl-long.447.

[41] S. Dathathri, A. Madotto, Lan, et al., Plug and Play Language Models: A Simple Approach to Controlled Text Generation, arXiv preprint arXiv:1912.02164 (2019).

[42] A. Liu, M. Sap, X. Lu, Swayamdipta, et al., DExperts: Decoding-Time Controlled Text Generation with Experts and Anti-Experts, in: C. Zong, F. Xia, W. Li, R. Navigli (Eds.), 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), Association for Computational Linguistics, Online, 2021, pp. 6691–6706. URL: https://aclanthology.org/2021.acl-long.522.

[43] Z. Tang, K. Zhou, Wang, et al., Detoxify Language Model Step-by-Step, arXiv preprint arXiv:2308.08295 (2023).

[44] Z. Gou, Z. Shao, Gong, et al., Critic: Large Language Models Can Self-Correct with Tool-Interactive Critiquing, arXiv preprint arXiv:2305.11738 (2023).

[45] F. Faal, K. Schmitt, J. Y. Yu, Reward Modeling for Mitigating Toxicity in Transformer-Based Language Models, Applied Intelligence 53 (2022) 8421–8435.

[46] X. Lu, S. Welleck, Hessel, et al., Quark: Controllable Text Generation with Reinforced [Un]learning, 36th Conference on Neural Information Processing Systems (NeurIPS 2022) (2022).

[47] S. Welleck, X. Lu, West, et al., Generating Sequences by Learning to Self-Correct, arXiv preprint arXiv:2211.00053 (2022).

[48] T. He, J. Glass, Detecting egregious responses in neural sequence-to-sequence models, arXiv preprint arXiv:1809.04113 (2018).

[49] E. Perez, S. Huang, F. Song, Cai, et al., Red Teaming Language Models with Language Models, in: Y. Goldberg, Z. Kozareva, Y. Zhang (Eds.), Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2022, pp. 3419–3448. URL: https://aclanthology.org/2022.emnlp-main.225.

[50] N. Ousidhoum, X. Zhao, Fang, et al., Probing Toxic Content in Large Pre-Trained Language 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), 2021, pp. 4262–4274.

[51] L. Ouyang, J. Wu, Jiang, et al., Training language models to follow instructions with human feedback, 2022, URL https://arxiv. org/abs/2203.02155 13 (2022).

[52] N. Lambert, L. von Werra, Illustrating Reinforcement Learning from Human Feedback (RLHF), Hugging Face Blog (2022).

[53] J. Shlens, Notes on Kullback-Leibler Divergence and Likelihood, arXiv preprint arXiv:1404.2000 (2014).

[54] H. Kim, Y. Yu, Jiang, et al., PROSOCIALDIALOG: A Prosocial Backbone for Conversational Agents, arXiv preprint arXiv:2205.12688 (2022).

[55] J. Xu, A. Szlam, J. Weston, Beyond Goldfish Memory: Long-Term Open-Domain Conversation, in: S. Muresan, P. Nakov, A. Villavicencio (Eds.), Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Dublin, Ireland, 2022, pp. 5180–5197. URL: https://aclanthology.org/2022.acl-long.356.

doi:10.18653/v1/2022.acl-long.356.

[56] Y. Bai, S. Kadavath, Kundu, et al., Constitutional AI: Harmlessness from AI Feedback, arXiv preprint arXiv:2212.08073 (2022).

[57] R. Ramamurthy, P. Ammanabrolu, Brantley, et al., Is Reinforcement Learning (Not) for Natural Language Processing: Benchmarks, Baselines, and Building Blocks for Natural Language Policy Optimization, arXiv preprint arXiv:2210.01241 (2022).

[58] Z. Zhao, Z. Zhang, F. Hopfgartner, A Comparative Study of Using Pre-trained Language Models for Toxic Comment Classification, in: Companion Proceedings of the Web Conference 2021, 2021, pp. 500–507.

[59] G. Song, D. Huang, Z. Xiao, A Study of Multilingual Toxic Text Detection Approaches under Imbalanced Sample Distribution, Information 12 (2021) 205.

[60] R. Dey, F. M. Salem, Gate-Variants of Gated Recurrent Unit (GRU) Neural Networks, in: 2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS), IEEE, 2017, pp. 1597–1600.

[61] M. Schuster, K. K. Paliwal, Bidirectional recurrent neural networks, IEEE Trans. Signal Process. 45 (1997) 2673–2681. URL: https://api.semanticscholar.org/CorpusID:18375389.

[62] J. Devlin, M.-W. Chang, Lee, et al., BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, in: J. Burstein, C. Doran, T. Solorio (Eds.), 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), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. URL: https://aclanthology.org/N19-1423. doi:10.18653/v1/N19-1423.

[63] Y. Liu, M. Ott, N. Goyal, Du, et al., RoBERTa: A Robustly Optimized BERT Pretraining Approach, arXiv preprint arXiv:1907.11692 (2019).

[64] P. He, X. Liu, J. Gao, W. Chen, DeBERTa: Decoding-enhanced BERT with Disentangled Attention, arXiv preprint arXiv:2006.03654 (2020).

[65] cjadams, J. Sorensen, J. Elliott, L. Dixon, et al., Toxic Comment Classification Challenge, 2017. URL: https://kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge.

[66] A. Papasavva, S. Zannettou, E. De Cristofaro, Stringhini, et al., Raiders of the Lost Kek: 3.5 Years of Augmented 4chan Posts from the Politically Incorrect Board, in: Proceedings of the international AAAI conference on web and social media, volume 14, 2020, pp. 885–894.

[67] Y. Chang, X. Wang, J. Wang, Wu, et al., A Survey on Evaluation of Large Language Models, arXiv preprint arXiv:2307.03109 (2023).

[68] S. Ghazarian, N. Wen, A. Galstyan, N. Peng, DEAM: Dialogue Coherence Evaluation using AMR-based Semantic Manipulations, in: S. Muresan, P. Nakov, A. Villavicencio (Eds.), Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Dublin, Ireland, 2022, pp. 771–785. URL: https://aclanthology.org/2022.acl-long.57. doi:10.18653/v1/2022.acl-long.57.

[69] W. Zhu, S. Bhat, GRUEN for Evaluating Linguistic Quality of Generated Text, in: T. Cohn, Y. He, Y. Liu (Eds.), Findings of the Association for Computational Linguistics: EMNLP 2020, Association for Computational Linguistics, Online, 2020, pp. 94–108. URL: https://aclanthology.org/2020.findings-emnlp.9. doi:10.18653/v1/2020.findings-emnlp.9.