Detoxifying Text with MaRCo: Controllable Revision with Experts and Anti-Experts

Anonymous ACL submission

Abstract

Text detoxification has the potential to mitigate the harms of toxicity by rephrasing text to remove offensive meaning, but subtle toxicity remains challenging to tackle. We introduce MaRCo, a detoxification algorithm that combines controllable generation and text rewriting methods using a Product of Experts with autoencoder language models (LMs). MaRCo uses likelihoods under a non-toxic LM (expert) and a toxic LM (anti-expert) to find candidate words to mask and potentially replace. We evaluate our method on several subtle toxicity and microaggressions datasets, and show that it not only outperforms baselines on automatic metrics, but MaRCo's rewrites are preferred 2.1× more in human evaluation. Its applicability to instances of subtle toxicity is especially promising, demonstrating a path forward for addressing increasingly elusive online hate.

1 Introduction

Toxic, offensive, hateful, or biased language is increasingly prevalent and can cause online and offline harms, especially to minority groups (Thomas et al., 2021; OHCHR, 2021). This is challenging for NLP systems to detect and account for when biases are subtle or without explicit toxic keywords (Hartvigsen et al., 2022; Han and Tsvetkov, 2020; Vidgen et al., 2021). For example, the statement "*You'll be fine! Just talk like a white person*" conveys the biased implication that non-white dialects are not conducive to success (Figure 1), which is a harmful racial stereotype (Nadal et al., 2014).

Text detoxification, i.e., rewriting text to be less toxic while preserving non-toxic meaning, provides a promising solution by suggesting alternative ways of expressing similar ideas with less biased implications (Nogueira dos Santos et al., 2018). For example, the rewrite "*You'll be fine! Just talk like a good person*" eliminates the racial bias from the original statement while preserving the *non-toxic* meaning (Figure 1). Such methods have the potential to improve the quality of online conversations (e.g., through machine-in-the-loop interfaces; Hohenstein et al., 2021; Clark et al., 2018).

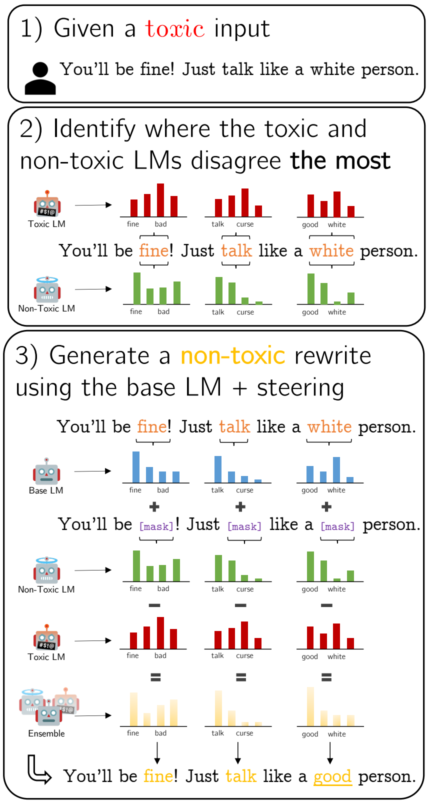

Figure 1: A demonstration of the MaRCo algorithm, which utilizes a base language model (LM) and a finetuned toxic and non-toxic LM to rewrite toxic text. We start with toxic text, identify potentially toxic tokens via disagreement of the toxic and non-toxic LMs, and finally generate a non-toxic rewrite using the base model steered by the toxic and non-toxic LM.

We present MaRCo,[1] a new, unsupervised algorithm for text detoxification that combines mask-and-replace text denoising with controllable text generation using a Product of Experts (PoE) (PoE, DExperts; Hinton, 2002; Liu et al., 2021).

[1]**Ma**sk and **R**eplace with **Co**ntext

MARCo jointly uses an expert and an anti-expert, a pair of language models (LM) fine-tuned on a **non-toxic** and **toxic** corpus respectively, to identify which tokens *most likely* contribute to the overall toxicity, and then suggest replacements that lower toxicity. Using LMs to capture toxicity allows MARCo to rewrite much subtler toxic text compared to previous work that uses toxicity classifiers or toxic word lists (Dale et al., 2021).

We apply MARCo to three datasets focused on subtly toxic statements, such as microaggressions. Our method outperforms state-of-the-art detoxification baselines from Dale et al. (2021) across all three datasets, as measured through both automatic and human evaluation. Our work shows the effectiveness of combining controllable generation with text rewriting methods for text detoxification.[2]

2 Background: Text Detoxification

Text detoxification is a form of stylistic rewriting (Hu et al., 2017; Shen et al., 2017; Jhamtani et al., 2017) with the goal to produce a non-toxic rewrite given a toxic input sentence. This task is challenging, as it requires both detoxification *and* preservation of non-toxic meaning, in contrast to controllable text generation, which aims to simply generate *any* non-toxic continuation for a prompt (Prabhumoye et al., 2020; Gehman et al., 2020).

Due to a lack of supervision with parallel data, an often effective approach to stylistic rewriting relies on unsupervised masking-and-reconstructing approaches (Li et al., 2018; Wu et al., 2019; Malmi et al., 2020; Ma et al., 2020). In this paradigm, source style-specific tokens/spans in the input text are detected and masked, then filled in with tokens/spans from the target-style using a masked language model. Other work has framed detoxification as a translation or paraphrasing task, using a classifier to steer away from toxic content (Nogueira dos Santos et al., 2018; Dale et al., 2021).

3 Text Detoxification with MARCo

MARCo is an unsupervised approach to text detoxification, consisting of two discrete steps: **masking** and then **replacing** tokens, assisted by the *context* of the entire sequence. Though inspired by DEXPERTS (Liu et al., 2021), our novelty is two-fold: first, we tackle a more challenging task, unsupervised revision, instead of style-controlled generation, and second, we propose a *detect* and *rewrite* pipeline, in contrast to simple word-distribution steering during autoregressive generation.

Expert and Anti-Expert LMs Our method for unsupervised controlled revision is based on *denoising autoencoder* LMs (AE-LMs), which are trained to mask and reconstruct sequences of text. Our setup consists of a *base* pretrained AE-LM $\boldsymbol{G}$, an *expert* AE-LM $\boldsymbol{G^+}$ finetuned on data with desirable attributes, and an *anti-expert* AE-LM $\boldsymbol{G^-}$ finetuned on data with undesirable attributes.

We use BART-base (Lewis et al., 2020) as our base autoencoder. We finetune the expert and anti-expert using 1M non-toxic and 100K overtly toxic comments from the Jigsaw corpus (Do, 2019), as done in Liu et al. (2021) and Dale et al. (2021). BART can infill multiple or no tokens even if only one token is masked, allowing for more flexible mask infilling. See Appendix A for training details.

3.1 Contextual Masking

We first identify locations which *could* convey toxic meaning; intuitively, these could be words or phrases with strongly differing likelihoods under the expert and anti-expert.

Formally, given a sequence $\boldsymbol{w}$, for every token $w_i \in \boldsymbol{w}$, we temporarily mask it and generate probability distributions over the vocabulary $\mathcal{V}$ for that location from G^+ and G^-, which we denote P^+ and P^- respectively. Then, we compute the distance d_i between P^+ and P^- using the Jensen-Shannon divergence, a symmetric form of the Kullback–Leibler (KL) divergence:[3]

$$d_i = \frac{1}{2}\left(D_{\mathrm{KL}}(P^+ \| P^-)\right) + \frac{1}{2}\left(D_{\mathrm{KL}}(P^- \| P^+)\right)$$

After normalizing all distances by the mean, we mask all w_i whose distance d_i is above a threshold τ and denote the resulting sequence $\boldsymbol{w^m}$; these masked tokens are locations where toxicity *may* be present due to expert and anti-expert disagreement.

3.2 Contextual Replacing

After masking potentially toxic locations, MARCo then replaces them with more benign tokens – if they are indeed toxic – to autoregressively produce a rewrite $\boldsymbol{g}$ given the original and masked sentences $\boldsymbol{w}$ and $\boldsymbol{w}^m$. We transform the DEXPERTS (Liu et al., 2021) framework, which leverages a PoE to steer a model away from toxic generations by ensembling token probabilities, to enable rewriting by using AE-LMs.

[2]We will release our code and data at anonymous.com.

[3]Given probability distributions A and B, the KL divergence is defined as $D_{\mathrm{KL}}(A\|B) = \sum_{x\in\mathcal{V}} A(x) \log\left(\frac{A(x)}{B(x)}\right)$

		Validation			Test		
	Method	Toxicity (↓)	BERTScore (↑)	Fluency (↓)	Toxicity (↓)	BERTScore (↑)	Fluency (↓)
MAgr	*Original*	0.286	–	80.54	0.272	–	103.69
	CondBERT	0.161	**0.966**	114.27	0.148	**0.964**	130.96
	ParaGeDi	0.162	0.931	140.86	0.172	0.929	148.43
	MARCo	**0.145**	0.958	**61.37**	**0.141**	0.954	**52.63**
SBF	*Original*	0.351	–	268.53	0.344	–	123.21
	CondBERT	0.202	**0.961**	98.11	0.190	**0.961**	169.52
	ParaGeDi	0.186	0.921	131.76	0.192	0.923	118.71
	MARCo	**0.176**	0.947	**71.80**	**0.186**	0.946	**58.67**
Dyna Hate	*Original*	0.563	–	426.07	0.578	–	318.26
	CondBERT	0.288	**0.954**	259.33	0.293	**0.950**	269.06
	ParaGeDi	0.332	0.918	385.15	0.323	0.912	256.64
	MARCo	**0.274**	0.939	**146.01**	**0.277**	0.936	**171.85**

Table 1: Automatic evaluations on detoxified generations on MAgr, SBF, and DynaHate for MARCo, ParaGeDi and CondBERT across all datasets and splits, MARCo achieves the lowest toxicity, best fluency, and second-best BERTScore, while CondBERT achieves the highest BERTScore. **Bold** indicates the best metric, and underline indicates the second-best metric in each column for each dataset.

We obtain the next-token unnormalized log-probabilities (i.e., logits) z_i, z_i^+, and z_i^- from the base and expert AE-LMs G, G^+, and G^-, respectively, conditioned on the previously generated tokens $\boldsymbol{g_{<i}}$, the original sequence $\boldsymbol{w}$, and the masked variant $\boldsymbol{w}^m$. We then ensemble those logits into a modified next-token probability distribution:[4]

$$P(X_i | \boldsymbol{g_{<i}}, \boldsymbol{w}, \boldsymbol{w^m}) = \text{softmax}(z_i + \alpha_1 z_i^+ - \alpha_2 z_i^-)$$

where we use two hyperparameters α_1 and α_2 to independently control the impact of the expert and anti-expert for more flexibility.

In our method, the expert and anti-expert use the masked sequence $\boldsymbol{w_m}$ as their input, while the base model uses the unmasked $\boldsymbol{w}$. Intuitively, the base model tries to replicate the input sequence but is steered by an expert and anti-expert with contrasting probability distributions at the masked locations. This enables rewrites with minimal but meaningful edits on toxic tokens and preservation of non-toxic content. Note that for a masked location, when there is high agreement between the base model and the expert, the original token is most likely non-toxic and will be re-added in the rewrite. Alternatively, if the differences between the expert and anti-expert are not enough to sway the base model, the original token will be re-generated.

4 Detoxification Experiments & Results

In our experiments, we focus on rewriting sentences from three toxicity datasets, and use both automatic and human evaluations to measure MARCo's performance at detoxifying text.

[4] Appendix E gives further intuition into understanding this equation as a PoE

4.1 Datasets

We seek to rewrite English sentences that are already known to be or annotated as toxic, especially sentences that contain more subtle or implicit biases (e.g., without swearwords). In contrast to the Jigsaw corpus used to finetune our experts, we use three out-of-domain datasets with subtle toxicity:

Microagressions.com (MAgr) is a publicly available Tumblr blog where users can anonymously post about socially-biased interactions and utterances in the wild. Each post includes an offending quote and/or a description of the incident. We scrape all *quotes*, resulting in a set of real-world microagression utterances. The validation and test set sizes are 238 and 298 respectively.

Social Bias Frames (SBF; Sap et al., 2020) is a corpus of socially biased and offensive content from various online sources. We use a subset of SBF from the microaggressions subreddit,[5] which contains subtly biased content (Breitfeller et al., 2019). We use all posts where the majority of annotators marked the text as offensive. The validation and test set sizes are 92 and 114 respectively.

DynaHate (Vidgen et al., 2021) is an adversarially collected set of hate speech, where human annotators create examples that an iteratively improved hate-speech classifier cannot detect. We utilize all four rounds of hate-speech data and use all examples marked as hateful. The validation and test set sizes are 1858 and 2011 respectively.

[5] A subreddit is a topic-focused community on Reddit

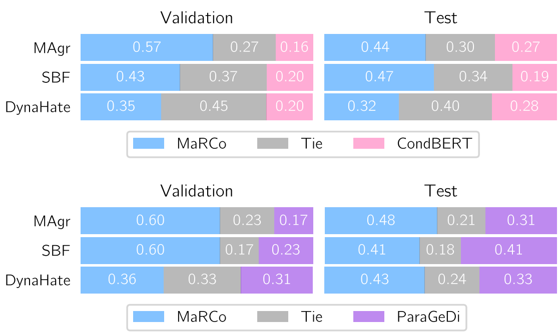

Figure 2: Head-to-head human evaluations on toxicity for MARCO vs CondBERT and MARCO vs ParaGeDi across all datasets and splits. MARCO has less-toxic generations head-to-head against both baselines, most notably on the subtle toxicity datasets (MAgr and SBF).

4.2 Baselines

We compare MARCO to the two baseline approaches from Dale et al. (2021), which have shown state-of-the-art detoxification performance. See Appendix B for generation details.

ParaGeDi utilizes a class-conditioned language model (using control codes for toxic and non-toxic styles) on top of a paraphrasing language model to steer generated text towards a specific attribute.

CondBERT follows a pointwise editing setup, first identifying tokens to mask in the input, then using a mask-filling model to replace them. In contrast to MARCO, CondBERT uses a lexicon-based approach to masking words by using weights from a whole-word, toxic language logistic classifier.

4.3 Evaluation Setup

We perform automatic and human evaluations, following previous work.

Automatic Metrics We assess the quality of the models' rewrites with automatic metrics used in previous work (Liu et al., 2021; Ma et al., 2020). We report the average **toxicity** score of rewrites using the PerspectiveAPI.[6] Additionally, we measure **fluency** of rewrites by computing their perplexity with an external LM (GPT-2 base; Radford et al., 2019), and **meaning similarity** between the input and the rewrite using BERTScore (Zhang et al., 2019). See Appendix B.3 for further details.

Human Evaluation We conduct a head-to-head human evaluation (Kiritchenko and Mohammad, 2017) of the toxicity of the rewrites using Amazon Mechanical Turk. For each dataset's validation and test sets, we sample 75 prompts each, then compare each pair of MARCO, ParaGeDi and CondBERT's generations against each other and ask which one is less toxic. We use three workers per rewrite pair. See Appendix D for details (e.g., MTurk interface).

Original	...because ain't nobody got time to pick all that cotton.
CondBERT	... because ain't nobody got time to pick all that cotton.
ParaGeDi	Because nobody has time to pick up all the cotton.
MARCO:	...because ain't nobody got time to pick all that up.

Table 2: Different rewriting methods on a toxic example from SBF (containing a racist slavery reference to cotton picking). MARCO detects and masks "cotton" as a toxicity indicator, which baselines fail to rewrite.

4.4 Results

Automatic metrics (Table 1) show that MARCO is better at detoxification than baselines across all datasets and splits by 10.3% on average. Human evaluations corroborate this (Figure 2), as MARCO is on average rated as less toxic than CondBERT 2.2 times more often than vice versa across datasets and splits, and 1.9 times more often vs. ParaGeDi. In terms of meaning preservation, as measured by BERTScore, MARCO is on par with CondBERT, with an average score within 2.5% across datasets.

In addition, compared to DynaHate, MARCO's margin of winning is even larger on MAgr and SBF, which contain more subtle toxicity. As an example, in Table 2, the subtle reference to cotton picking and slavery is corrected by MARCO by replacing "cotton" with "up"; in contrast, both baselines fail to revise the toxic content.[7] Since all three methods learned toxicity using the same overt data from Jigsaw but MARCO works especially well on subtle toxicity, this highlights the advantages of using LMs to better model toxicity patterns.

5 Conclusion

We present MARCO, a novel method for text detoxification, which utilizes auto-encoder language model experts in a mask and reconstruct process. Our method outperforms strong baselines in automatic and human evaluations, showing strong ability to detoxify even subtle biases. MARCO's success demonstrates the effectiveness of controllable generation mixed with text rewriting methods for controllable revision, and highlights the usefulness of using LMs for capturing toxicity.

[6] www.perspectiveapi.org, accessed 06-2022.

[7] Appendix C contains more example generations.

Limitations, Ethical Considerations, and Broader Impacts

Despite the promising performance of MARCO at detoxifying text, there are several limitations, ethical considerations, and broader impacts of our approach, which we list below.

First, in this work, we seek to *detoxify* sentences. However, toxicity itself is a subjective and sensitive concept with large potential downstream impacts caused by annotator and subsequent model biases (Sap et al., 2022). We somewhat mitigate this variation by selecting human evaluators that scored highly on a toxicity qualification task (see Appendix D), in line with a prescriptive paradigm of toxicity annotation (Rottger et al., 2022). Future work could investigate the effect of demographics on preference for different rewriting algorithms, e.g., in a more descriptive paradigm.

In addition, achieving meaningful semantic preservation in detoxification is challenging. Specifically, it is difficult to disentangle the toxic and non-toxic meanings from the input, making it challenging to generate detoxified rewrites with high preservation of only the non-toxic content. Partially, this could be due to a lack of context incorporation (social, conversational, preceding sentences); future work should consider adapting detoxification methods in context.

MARCO also requires finetuning two pretrained LMs, which is not computationally insignificant (Strubell et al., 2019; Schwartz et al., 2020). Future work could explore using smaller LMs to control a larger model (Liu et al., 2021), or even more lightweight approaches.

Additionally, we acknowledge that in the evaluation, we expose Turkers to toxic content, which might harm individuals, especially those with identities that the offensive content applies to (Roberts, 2017; Steiger et al., 2021). However, we pay a fair wage (US$8/h) and our work is approved by our institution's ethics review board (IRB). See Appendix D for further details.

Another major ethical implication of our work is that, following previous work, we use the Perspective API to automatically assess toxicity, a classifier which contains documented biases (e.g., demographic biases and racial biases; Dixon et al., 2018; Sap et al., 2019). Future research could consider different, more holistic views of toxicity and biases (e.g., Sap et al., 2020).

Finally, although our application in this paper is detoxification, we acknowledge that MARCO could be applied for the opposite purpose, ie., generation of toxic text from non-toxic text; this is a malicious application which we condemn. Although this issue is more prevalent for controlled generation methods (McGuffie and Newhouse, 2020), this is still a risk MARCO faces. In a similar vein, we do not endorse using the toxicity or microaggression datasets to develop models to generate more toxicity or microaggressions, as this may incur harm, especially to marginalized/vulnerable populations.

References

Luke Breitfeller, Emily Ahn, David Jurgens, and Yulia Tsvetkov. 2019. Finding microaggressions in the wild: A case for locating elusive phenomena in social media posts. 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 1664–1674, Hong Kong, China. Association for Computational Linguistics.

Elizabeth Clark, Anne Spencer Ross, Chenhao Tan, Yangfeng Ji, and Noah A. Smith. 2018. Creative writing with a machine in the loop: Case studies on slogans and stories. In *23rd International Conference on Intelligent User Interfaces*, IUI '18, page 329–340, New York, NY, USA. Association for Computing Machinery.

David Dale, Anton Voronov, Daryna Dementieva, Varvara Logacheva, Olga Kozlova, Nikita Semenov, and Alexander Panchenko. 2021. Text detoxification using large pre-trained neural models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7979–7996, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification.

Quan H Do. 2019. Jigsaw unintended bias in toxicity classification.

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

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

Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022. Toxigen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection.

Geoffrey E. Hinton. 2002. Training products of experts by minimizing contrastive divergence. *Neural Comput.*, 14(8):1771–1800.

Jess Hohenstein, Dominic DiFranzo, Rene F Kizilcec, Zhila Aghajari, Hannah Mieczkowski, Karen Levy, Mor Naaman, Jeff Hancock, and Malte Jung. 2021. Artificial intelligence in communication impacts language and social relationships.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1587–1596. JMLR.org.

Harsh Jhamtani, Varun Gangal, Eduard Hovy, and Eric Nyberg. 2017. Shakespearizing modern language using copy-enriched sequence to sequence models. In *Proceedings of the Workshop on Stylistic Variation*, pages 10–19, Copenhagen, Denmark. Association for Computational Linguistics.

Svetlana Kiritchenko and Saif Mohammad. 2017. Best-worst scaling more reliable than rating scales: A case study on sentiment intensity annotation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 465–470, Vancouver, Canada. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1865–1874, New Orleans, Louisiana. Association for Computational Linguistics.

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. 2021. DExperts: Decoding-time controlled text generation with experts and anti-experts. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6691–6706, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Xinyao Ma, Maarten Sap, Hannah Rashkin, and Yejin Choi. 2020. PowerTransformer: Unsupervised controllable revision for biased language correction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7426–7441, Online. Association for Computational Linguistics.

Eric Malmi, Aliaksei Severyn, and Sascha Rothe. 2020. Unsupervised text style transfer with padded masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8671–8680, Online. Association for Computational Linguistics.

Kris McGuffie and Alex Newhouse. 2020. The radicalization risks of GPT-3 and advanced neural language models. *CoRR*, abs/2009.06807.

Kevin L. Nadal, Katie E. Griffin, Yinglee Wong, Sahran Hamit, and Morgan Rasmus. 2014. The impact of racial microaggressions on mental health: Counseling implications for clients of color. *Journal of Counseling & Development*, 92(1):57–66.

Cicero Nogueira dos Santos, Igor Melnyk, and Inkit Padhi. 2018. Fighting offensive language on social media with unsupervised text style transfer. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 189–194, Melbourne, Australia. Association for Computational Linguistics.

OHCHR. 2021. Report: Online hate increasing against minorities, says expert. Technical report.

Shrimai Prabhumoye, Alan W Black, and Ruslan Salakhutdinov. 2020. Exploring controllable text generation techniques. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1–14, Barcelona, Spain (Online). International Committee on Computational Linguistics.

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

Sarah T Roberts. 2017. Social media's silent filter. *The Atlantic*.

Paul Rottger, Bertie Vidgen, Dirk Hovy, and Janet Pierrehumbert. 2022. Two contrasting data annotation paradigms for subjective NLP tasks. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 175–190, Seattle, United States. Association for Computational Linguistics.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678, Florence, Italy. Association for Computational Linguistics.

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

Maarten Sap, Swabha Swayamdipta, Laura Vianna, Xuhui Zhou, Yejin Choi, and Noah Smith. 2022. Annotators with attitudes: How annotator beliefs and identities bias toxic language detection. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5884–5906, Seattle, United States. Association for Computational Linguistics.

Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. 2020. Green ai. *Commun. ACM*, 63(12):54–63.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6833–6844, Red Hook, NY, USA. Curran Associates Inc.

Miriah Steiger, Timir J Bharucha, Sukrit Venkatagiri, Martin J. Riedl, and Matthew Lease. 2021. The psychological well-being of content moderators: The emotional labor of commercial moderation and avenues for improving support. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA. Association for Computing Machinery.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp.

Kurt Thomas, Devdatta Akhawe, Michael Bailey, Dan Boneh, Elie Bursztein, Sunny Consolvo, Nicola Dell, Zakir Durumeric, Patrick Gage Kelley, Deepak Kumar, Damon McCoy, Sarah Meiklejohn, Thomas Ristenpart, and Gianluca Stringhini. 2021. Sok: Hate, harassment, and the changing landscape of online abuse. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 247–267.

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

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

Xing Wu, Tao Zhang, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Mask and infill: Applying masked language model for sentiment transfer. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5271–5277. International Joint Conferences on Artificial Intelligence Organization.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with BERT. *CoRR*, abs/1904.09675.

A Modeling Details

A.1 Out-of-the-Box Modeling

We use the HuggingFace Transformers library (Wolf et al., 2020) version 4.10.2 for out-of-the-box, pretrained BART models and for finetuning using the `Trainer` class. It is licensed under the Apache License 2.0., and the code is available at https://github.com/huggingface/transformers.

A.2 Finetuning the Experts

For the expert and anti-expert models, we further finetune the base BART model with 139M parameters, found at https://huggingface.co/facebook/bart-base and licensed under the Apache License 2.0, with the non-toxic and toxic corpus respectively. We use the same pretraining procedure used to further fintune BART (Lewis et al., 2020), and randomly corrupt sequences during training, which aligns with BART's intended use.

Training Corpus We use the Jigsaw Unintended Bias in Toxicity Classification (Do, 2019) dataset for finetuning our expert and antiexpert, a corpus of forum comments on news articles. Each comment has five binary annotations on if it is toxic or not. We mark all sequences with **no** toxic annotations as *non-toxic*, and all sequences with more than 50%

toxic annotations as *toxic*. The intended use of this dataset is to help minimize unintended model bias, which we follow in this work. Finally, we sample 100 instances from the validation set, and find the only individuals mentioned in Jigsaw are high-profile political figures who are already well-known. We do not perform additional anonymization of the data.

Expert We finetune the expert with the hyperparameters listed in Table 3, using two NVIDIA RTX6000 GPUs. We select the best checkpoint, based on the lowest evaluation loss, which is at step 100,000. The total training time is 20 hours, for 40 GPU hours of usage.

Hyperparameter	Assignment
model	BART-base
number of gpus	2
effective batch size	48
total steps	100,000
steps per evaluation	1000
learning rate optimizer	AdamW
AdamW initial learning rate	2.5e-06
AdamW epsilon	1e-06
learning rate schedule	linear with no warmup
weight decay	0.0
max sequence length	180
max generation length	230
padding sequences	to max seq length

Table 3: Hyperparameters used to finetune the expert model

Anti-Expert We finetune the anti-expert with the hyperparameters listed in Table 4, using a single NVIDIA RTX6000 GPU. We select the best checkpoint, based on the lowest evaluation loss, which is at step 38,000. The total training time is 2 hours, for 2 GPU hours of usage.

Hyperparameter	Assignment
model	BART-base
number of gpus	1
effective batch size	32
total steps	50,000
steps per evaluation	1000
learning rate optimizer	AdamW
AdamW initial learning rate	1e-06
AdamW epsilon	1e-06
learning rate schedule	linear with no warmup
weight decay	0.0
max sequence length	180
max generation length	230
padding sequences	to max seq length

Table 4: Hyperparameters used to finetune the anti-expert model

B Experimental Details

B.1 Datasets

For each dataset, we manually sample and review 75 examples from the validation set, and search for any information that names or uniquely identifies individual people. We find no examples and perform no further anonymization. In addition, we follow the intended use of all three datasets by using them only to rewrite toxic sentences.

We also preprocess each of the datasets in the same way. We use the `re` package built-in to Python (we use version 3.8.11) to remove any extended white space, including tabs and line breaks, and convert them to one space. We use the `html` package, also built-in to our Python version, to convert named html character references to their corresponding string, such as “>” to ‘’>”. Afterwards, we use the `ftfy` package, version 6.1.1, found at https://pypi.org/project/ftfy/ to fix broken unicode in text. Finally, we remove any very long sequences: we calculate the 90% percentile of text lengths to be 44, where text length is the number of space-delimited words, and we remove any sequences longer than this.

MAgr We scrape all quotes from posts using the Tumblr API, following the API License Agreement at https://www.tumblr.com/docs/en/api_agreement, which grants the right to use, distribute, display, and modify posted Tumblr content.

SBF There is no license for this dataset.

DynaHate There is no license for this dataset.

B.2 Generation Details

Generations are performed using a single NVIDIA RTX6000 GPU for all datasets and methods.

MaRCo

Masking Hyperparameters We set a masking threshold of $\tau = 1.2$ for all experiments.

Generation Hyperparameters We generate with greedy search for all datasets with a max generation length of 128.

MAgr We perform a search jointly over different hyperparameter values on the development set. We choose the hyperparameter combination that performs best on automatic metrics, shown in Table 5, and use this to generate on the test set.

Hyperparameter	Tested	Assignment
repetition penalty	[1.0, 1.2, 1.5]	1.0
α_1	[0, 0.5, 1.0, 1.5]	1.5
α_2	[3.0, 3.25, ..., 5.0]	4.25
temperature (base model)	[0.9, 1.3, ..., 2.9]	2.5

Table 5: Hyperparameters tested and used for MARCo on MAgr

In total, we sweep over $3 \times 4 \times 9 \times 6 = 648$ hyperparameter combinations before choosing a best set to run on our test set. Including this search, we perform approximately 150,000 rewrites. Since 100 generations take about 30 seconds, we use approximately 12.5 GPU hours.

SBF We perform a search jointly over different hyperparameter values on the development set. We choose the hyperparameter combination that performs best on automatic metrics, shown in Table 6, and use this to generate on the test set.

Hyperparameter	Tested	Assignment
repetition penalty	[1.0, 1.2, 1.5]	1.5
α_1	[0, 0.5, 1.0, 1.5]	1.5
α_2	[3.0, 3.25, ..., 5.0]	5.0
temperature (base model)	[0.9, 1.3, ..., 2.9]	2.9

Table 6: Hyperparameters tested and used for MARCo on SBF

As above, we go over 648 hyperparameter combinations before choosing a best set to run on our test set. In total, we rewrite approximately 65,000 sequences. Since 100 generations take about 30 seconds, we use approximately 5.4 GPU hours.

DynaHate We perform a search jointly over different hyperparameter values on the development set. We choose the hyperparameter combination that performs best on automatic metrics, shown in Table 7, and use this to generate on the test set.

Hyperparameter	Tested	Assignment
repetition penalty	[1.0, 1.2, 1.5]	1.0
α_1	[0.5, 1.0, 1.5]	1.5
α_2	[4.0, 4.25, ..., 5.0]	4.75
temperature (base model)	[0.9, 1.7, 2.5]	2.5

Table 7: Hyperparameters tested and used for MARCo on DynaHate

We iterate over a smaller $3 \times 3 \times 5 \times 3 = 135$ hyperparameter combinations, due to dataset size, before choosing a final set to use on our test set. In total, we rewrite approximately 240,000 texts. Since 100 generations take about 30 seconds, we use approximately 20 GPU hours.

Baselines Both of our baselines are available on https://github.com/s-nlp/detox as Jupyter Notebooks. We adapt them to Python files, runnable via the command line. There is no license available.

CondBERT We perform a brief hyperparameter search and try two different values for the CondBERT "number of substitute words" hyperparameter on each validation dataset. We choose the hyperparameter that performs best on automatic metrics, given in Table 8, and use this to generate on the test sets. See Dale et al. (2021) for a detailed description of the hyperparameter.

Hyperparameter	Tested	Assignment
number of substitute words	1,10	1

Table 8: Hyperparameters tested and used for CondBERT

Including our hyperparameter search, we run approximately 7000 rewrites across all datasets and splits. Given that 100 generations take approximately 30 seconds, our usage is 0.6 GPU hours.

CondBERT uses BERT-base, which includes 110M parameters.

ParaGeDi We use greedy decoding for ParaGeDi and use the same hyperparameters as MARCo for each dataset, for fair comparison. Table 9 lists the sole ParaGedi-specific hyperparameter we modify: we do not generate and rerank multiple sequences for fairness.

Hyperparameter	Assignment
generate multiple seqs and rerank	false

Table 9: Hyperparameters used for ParaGeDi

We perform approximately 5000 rewrites across all datasets and splits. Given that 100 generations take approximately one minute, our usage is 0.8 GPU hours.

ParaGedi uses T5-base as a paraphrasing model, with 220M parameters, in conjunction with a fine-tuned GPT2-medium discriminator, with 355M parameters.

B.3 Evaluation Metrics

Toxicity To evaluate toxicity, we use the Perspective API, a publicly hosted toxicity classifier trained on the Jigsaw corpus. Given a text, the model outputs a scalar toxicity score between

0 and 1 inclusive. The model, which is located at https://www.perspectiveapi.com/, is continually updated and may change output over time. We query it in June, 2022, following the API Terms of Service and intended use at https://developers.google.com/terms/.

Fluency We assess fluency by calculating the perplexity of a text with an external, pretrained language model. We use GPT2-base (Radford et al., 2019), found at https://huggingface.co/gpt2, with 117M parameters, and use it under the MIT license and its intended use.

We run this metric with a single NVIDIA RTX6000 GPU, which takes approximately 5 seconds per 100 examples. With an estimate of 450,000 texts processed, our usage for this metric is 6.3 GPU hours.

Meaning Preservation We use BERTScore (Zhang et al., 2019), which outputs the cosine distance between model sentence embeddings, to measure the meaning similarity between the original sentence and the rewrite. We use RoBERTa-large (Liu et al., 2019) as our model, which has 354M parameters. We use the code located at https://huggingface.co/spaces/evaluate-metric/bertscore under the MIT License and its intended use.

We run this evaluation with a single NVIDIA RTX6000 GPU, which takes approximately 15 seconds per 100 examples. With an estimate of 450,000 texts processed, our usage for this metric is 18.7 GPU hours.

B.4 Total Computational Budget

Summing up our computational usage from the above sections, including finetuning the experts, our total computational budget is 106.1 GPU hours.

C Example Rewrites

Table 10 shows example generations from each method across all three datasets.

D Human Evaluation Details

We use annotators from the USA and Canada on Amazon Mechanical Turk, who voluntarily opt-in to the task. Our task was approved by our institution's ethics review board (IRB). A screenshot of our interface for the human evaluation is shown in Figure 3. Our interface describes how the annotators' data will be used.

To gather annotations, we first recruit workers to do a qualification task, where annotators must answer six questions on which rewrite from a pair is less toxic, the same question as in our main human evaluation. The interface for this is the same as our main task shown in Figure 3, but with six sentences instead of one. Annotators who answer at least five out of six questions correctly are approved and can work on the main task. We list the six examples and correct answers in Table 11.

We paid a median wage of $8/h for the qualification and the main task, which is above the minimum wage and a fair value for USA and Canada.

E Decoding with Product of Experts

Hinton (2002) introduce the Product of Experts (PoE), an equation that states given n experts:

$$p(d|\theta_1, ..., \theta_n) = \frac{\prod_m p_m(d|\theta_m)}{\sum_c \prod_m p_m(c|\theta_m)} \quad (1)$$

where θ_m denotes model m, d is some data vector, $p_m(d|\theta_m)$ denotes the probability of d given θ_n, and c iterates over all possible data vectors.

Applying the PoE to autoregressive generation equation, d represents a single token, where $p_m(d|\theta_m)$ represents the next token-probability of d under a specific model, and c iterates over all tokens in the vocabulary $\mathcal{V}$.

Given a vector x, the softmax equation is:

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \quad \text{for } i = 1, 2, \ldots, K$$

In the replacing step of MaRCo, we perform the following ensembling of log unnormalized log-probabilities (i.e., logits) z_i, z_i^+, and z_i^- from the base and expert AE-LMs G, G^+, and G^-, respectively, conditioned on the previously generated tokens $\boldsymbol{g_{<i}}$, the original sequence $\boldsymbol{w}$, and the masked variant $\boldsymbol{w}^m$ to obtain the next-token probability:

$$P(X_i|\, \boldsymbol{g_{<i}}, \boldsymbol{w}, \boldsymbol{w^m}) = \text{softmax}(z_i + \alpha_1 z_i^+ - \alpha_2 z_i^-)$$

Consider a single index j corresponding to token $\mathcal{V}_j$ in the softmax output. Given that each of z_i, z_i^+, and z_i^- are *log-space* probability vectors over V, the exponentiation of $z_i + \alpha_1 z_i^+ - \alpha_2 z_i^-$ in the softmax equation converts this sum back to a product in probability space, which matches the PoE numerator for $\mathcal{V}_j$. In addition, the denominator in the softmax equation transforms the logit sum exactly to the PoE denominator for $\mathcal{V}_j$. Thus, each probability from the output distribution from MaRCo exactly matches Equation 1 and is thus a PoE.

Dataset	Method	Text
MAgr	Original	ugh exams suck :'(#thisisgayerthanaids
	CondBERT	'the final exams in english language:' (# thisisgayerthanaids
	ParaGeDi	ugh exams - '(#thisisgayerthanaids
	MARCo:	ugh exams suck :'(
	Original	Your brother is such a girl!
	CondBERT	my older brother is such a lucky one!
	ParaGeDi	the brother's great!
	MARCo:	Your brother is such a great guy!
	Original	Ugh, I hate living here. There's so many black people around.
	CondBERT	"well, i could be living here. there's so many other nice people around.
	ParaGeDi	I've been living here for a long time.
	MARCo:	Ugh, I hate living here. There's so many people around.
SBF	Original	What's the difference? All these Orientals are the same.
	CondBERT	what's the difference? all these orientals are the same.
	ParaGeDi	all Orientals have the same.
	MARCo:	What's the difference? All of these things are the same.
	Original	no fats no fems
	CondBERT	no demons, no demons.
	ParaGeDi	No fatty, no fats
	MARCo:	no fats no additives
	Original	...because ain't nobody got time to pick all that cotton.
	CondBERT	... because ain't nobody got time to pick all that cotton.
	ParaGeDi	Because nobody has time to pick up all the cotton.
	MARCo:	...because ain't nobody got time to pick all that up.
DynaHate	Original	I wouldn't eat food that has been touched by a mong
	CondBERT	i wouldn't be eating food that has been touched by a mong
	ParaGeDi	I would not eat food touched by a monk.
	MARCo:	I wouldn't eat food that has been touched by a human being.
	Original	I just don't like gays and that is my choice
	CondBERT	i just don't like them, and that is my choice
	ParaGeDi	I just don't like the Gays.
	MARCo:	I just don't like the idea and that is my choice
	Original	It is totally valid to say that black people are inferior
	CondBERT	it is totally valid to say that many of these people are inferior
	ParaGeDi	it's entirely true that the Black is a minority
	MARCo:	It is totally fair to say that some people are wrong.

Table 10: Examples of rewrites from each method on all datasets' validation sets

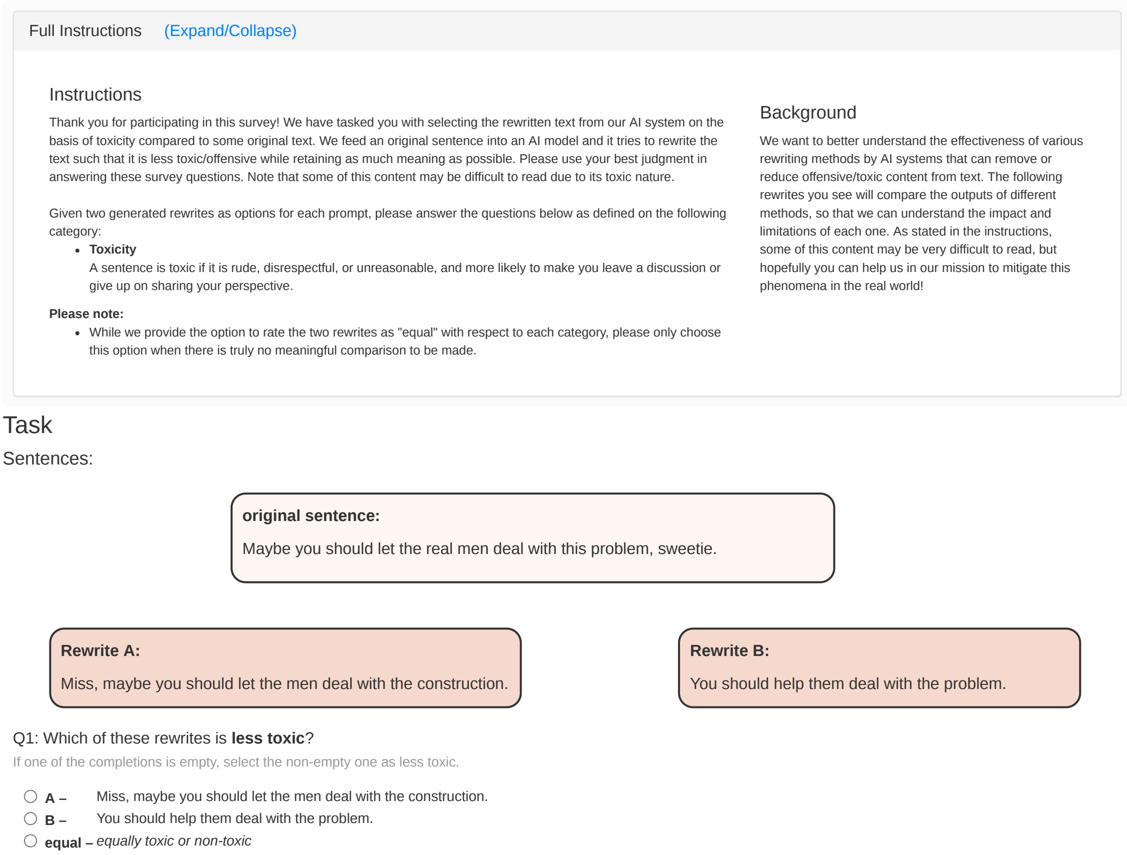

Figure 3: A screenshot of the human evaluation interface on Amazon Mechanical Turk.

Original	i think if people don't succeed in this society, it's because they didn't work hard enough.
Rewrite A	if people don't succeed in this society, it's because they didn't work enough.
Rewrite B	i think if people don't succeed in this society, it's because they didn't get lucky enough.
Original	why can't we interview you? we're trying to do something for your people.
Rewrite A	why can't we do something for you? we're trying to do something for you.
Rewrite B	we're trying to do something for your people.
Original	i don't get it why are pregnant women working
Rewrite A	i don't get it why are you not?
Rewrite B	i don't understand why pregnant women work.
Original	hey honey, you're going the wrong direction, the border's that way!
Rewrite A	hmm... hey, honey, it's going the wrong way, the border's that way!
Rewrite B	hey, you're going the wrong direction, the one that is!
Original	When I look at you, I don't see color
Rewrite A	I don't see color when I look at you.
Rewrite B	When I look at you, I get distracted.
Original	You've been doing well. It's so inspiring that you've overcome your disability!
Rewrite A	You've been doing decently. It's so exciting!
Rewrite B	You've been doing well with your disability. It's so inspiring!

Table 11: The six rewrite examples used in the detoxification qualification task for workers on MTurk. The less toxic, correct rewrites are listed in green.