DExperts: Decoding-Time Controlled Text Generation with Experts and Anti-Experts

Alisa Liu[♡] **Maarten Sap**[♡] **Ximing Lu**[♡♣] **Swabha Swayamdipta**[♣]
Chandra Bhagavatula[♣] **Noah A. Smith**[♡♣] **Yejin Choi**[♡♣]
[♡]Paul G. Allen School of Computer Science & Engineering, University of Washington
[♣]Allen Institute for Artificial Intelligence
alisaliu@cs.washington.edu

Abstract

Despite recent advances in natural language generation, it remains challenging to control attributes of generated text. We propose DExperts: Decoding-time Experts, a decoding-time method for controlled text generation that combines a pretrained language model with "expert" LMs and/or "anti-expert" LMs in a product of experts. Intuitively, under the ensemble, tokens only get high probability if they are considered likely by the experts and unlikely by the anti-experts. We apply DExperts to language detoxification and sentiment-controlled generation, where we outperform existing controllable generation methods on both automatic and human evaluations. Moreover, because DExperts operates only on the output of the pretrained LM, it is effective with (anti-)experts of smaller size, including when operating on GPT-3. Our work highlights the promise of tuning small LMs on text with (un)desirable attributes for efficient decoding-time steering.

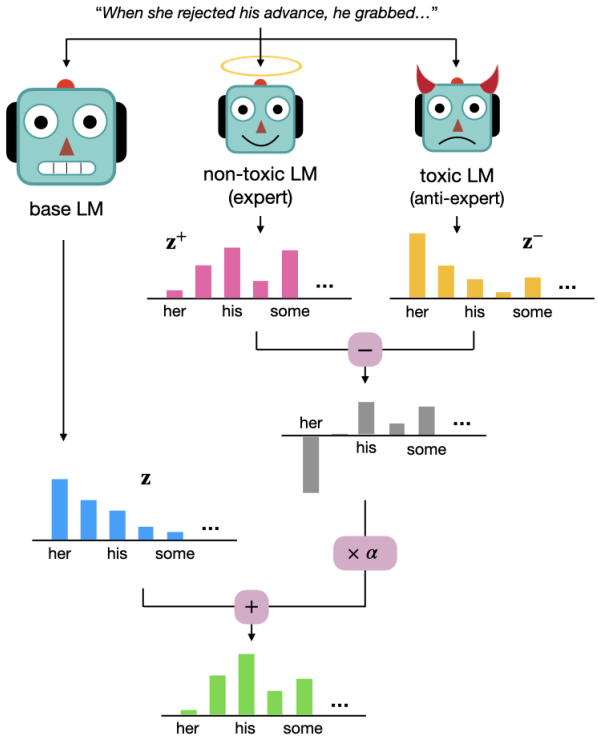

Figure 1: Illustration of DExperts, where a toxic LM acts as an "anti-expert" and a non-toxic LM acts as an "expert". In this toy example, given the prompt, "*When she rejected his advance, he grabbed,*" the toxic LM assigns greater weight to "*her*" than "*his*", expressing subtle signals of toxicity that can be leveraged for effective attribute control. The difference in logits $\mathbf{z}^+ - \mathbf{z}^-$ output by the expert and anti-expert represents the perturbations to make to the logits $\mathbf{z}$ of the pretrained "base" LM.

1 Introduction

Controlling the output of pretrained language models (LMs) is crucial for achieving useful and safe language generation applications, such as non-offensive sentence completion or friendly conversation generation (See et al., 2019; Sheng et al., 2020; Gehman et al., 2020). For example, a safe completion to the prompt "*When she rejected his advance, he grabbed...*" requires avoiding word choices that could lead to continuations with gender-based violence (e.g., "*her*"; Figure 1).

Without such steering, these language models risk generating mindless and offensive content (Sheng et al., 2019; Holtzman et al., 2020) which hinders their safe deployment (Brockman et al., 2020; Bender et al., 2021). Importantly, as the scale of pretrained LMs increases (e.g., 175B and 1.6T parameters; Brown et al., 2020; Fedus et al., 2021), finetuning or re-training approaches are becoming increasingly computationally infeasible for most researchers.

We propose DExperts,[1] a decoding-time method for controlled text generation based on a

[1]DExperts stands for Decoding-time Experts. Our code is available at https://github.com/alisawuffles/DExperts.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 6691–6706
August 1–6, 2021.

product of experts (Hinton, 2002). Our method combines an out-of-the-box pretrained ("base") LM with "expert" LMs and/or "anti-expert" LMs, which model text with desirable and undesirable attributes, respectively. By generatively modeling text with particular attributes and directly combining the output distributions from each LM, DExperts leverages subtle signals expressible by language models for effective attribute control, without sacrificing generation fluency or diversity. Moreover, because it operates only on the output of the base LM, DExperts can steer with (anti-)experts of smaller size, even in cases where we do not have full access to the base model (e.g., GPT-3 through an API).

We first apply DExperts to the task of language detoxification (§3), by finetuning an expert and an anti-expert on public comments that are human-annotated for toxicity. Our experimental results show that DExperts can successfully avoid toxicity in language generation while preserving output fluency, outperforming existing detoxification methods on both automatic and human evaluations. Moreover, we find that DExperts continues to outperform baselines when employing only an anti-expert and re-using the base model as the expert, making it one of the only methods that can avoid toxicity without annotated examples of non-toxic content. In analysis, we also show that our method successfully avoids toxic degeneration while using just ∼650 toxic comments, opening avenues for easily customizable anti-experts.

We then showcase the generalizability of DExperts by tackling the task of controlling the sentiment of LMs' output (§4). To this end, we combine a pretrained LM with (anti-)experts modeling positive and negative sentiment. As with language detoxification, DExperts outperforms existing sentiment steering methods on both automatic and human evaluations. Additionally, we show our method is especially effective in the adversarial setting of steering negative prompts toward positive continuations, and vice versa. Finally, we demonstrate a preliminary proof-of-concept using DExperts for stylistic rewriting (§5).

Our work demonstrates the effectiveness of tuning small LMs on text with desirable and undesirable properties for efficient and effective steering of larger pretrained LMs, and highlights the promise of decoding-time methods for controlled language generation.

2 Experts and Anti-Experts for Controlled Generation

Given input text as a **prompt**, the task of controlled text generation is to generate a **continuation** that flows naturally from the prompt while having the desired attribute (e.g., positive sentiment) but not an undesired one (e.g., toxicity).

Given a prompt $\boldsymbol{x}_{<t}$, the language model computes the logits for the tth token, denoted $\mathbf{z}_t \in \mathbb{R}^{|\mathcal{V}|}$, where $\mathcal{V}$ is the vocabulary. A probability distribution over the vocabulary is obtained by normalizing and exponentiating $\mathbf{z}_t$:

$$P(X_t \mid \boldsymbol{x}_{<t}) = \text{softmax}(\mathbf{z}_t), \tag{1}$$

and the next token is generated by sampling $x_t \sim P(X_t \mid \boldsymbol{x}_{<t})$.

2.1 DExperts Formalization

DExperts operates on a pretrained language model M by combining its predictions with an expert M^+, which models text with a desirable attribute, and an anti-expert M^-, which models text with an undesirable attribute. At time step t, we condition each language model M, M^+, and M^- on the prompt $\boldsymbol{x}_{<t}$ to obtain $\mathbf{z}_t, \mathbf{z}_t^+$, and $\mathbf{z}_t^-$, respectively. The product-of-experts ensemble is given by:[2]

$$\tilde{P}(X_t \mid \boldsymbol{x}_{<t}) = \text{softmax}\left(\mathbf{z}_t + \alpha\left(\mathbf{z}_t^+ - \mathbf{z}_t^-\right)\right) \tag{2}$$

where α is a hyperparameter that controls the amount of modification to $\mathbf{z}_t$, and can be interpreted as the strength of control over the base model. Equivalently,

$$\tilde{P}(X_t \mid \boldsymbol{x}_{<t}) \propto P(X_t \mid \boldsymbol{x}_{<t}) \left(\frac{P^+(X_t \mid \boldsymbol{x}_{<t})}{P^-(X_t \mid \boldsymbol{x}_{<t})}\right)^{\alpha} \tag{3}$$

Intuitively, a token will only have high probability if it has high probability under both P and P^+, and low probability under P^-. We can interpret the ratio $\frac{P^+(X_t|\boldsymbol{x}_{<t})}{P^-(X_t|\boldsymbol{x}_{<t})}$ as a scaling coefficient for each token, which is used to modify the original probability predicted for that token.

2.2 Sampling from DExperts

Sampling fluent output from language models commonly requires truncating the unreliable tail of

[2] Though not explored in this paper, this formulation readily accommodates multiple experts and anti-experts, whose logits can be respectively added or subtracted.

the probability distribution, as in top-k (Fan et al., 2018) or nucleus sampling (Holtzman et al., 2020). We adapt this intuition to our method by truncating the logits $\mathbf{z}$ output by the base model *prior* to combining with the experts. Formally, let $\mathcal{V}' \subset \mathcal{V}$ denote the set of tokens that are a part of the top-k/top-p vocabulary of the base LM at time step t. The truncated logits $\mathbf{z}'$ are given by

$$\mathbf{z}'[v] = \begin{cases} \mathbf{z}[v] & \text{if } v \in \mathcal{V}' \\ -\infty & \text{otherwise} \end{cases} \quad (4)$$

By substituting $\mathbf{z}$ with $\mathbf{z}'$ in Equation 2, we have

$$\tilde{P}'(X_t \mid \boldsymbol{x}_{<t}) = \text{softmax}\left(\mathbf{z}'_t + \alpha\left(\mathbf{z}_t^+ - \mathbf{z}_t^-\right)\right) \quad (5)$$

We obtain our next token x_t via *pure sampling* from the probability distribution $\tilde{P}'(X_t \mid \boldsymbol{x}_{<t})$, which has non-zero probability only on tokens in $\mathcal{V}'$. In this way, adding in the (anti-)experts can be interpreted as modifying the probability distribution over the candidate tokens in $\mathcal{V}'$, without any chance of reintroducing tokens $v \notin \mathcal{V}'$ from the tail of the original probability distribution.

3 Toxicity Avoidance

Given that large pretrained LMs are at risk of producing toxic content (Sheng et al., 2019; Gehman et al., 2020), steering away from toxic "degeneration" is crucial for their safe deployment. Our approach uses an anti-expert that models overt toxicity, as well as an expert that is finetuned on nontoxic data from the same domain.

Note that while obtaining an LM that is truly *free* from social biases is impossible (Fiske, 1993; Lakoff, 1973), the "non-toxic" expert serves the purpose of modeling the same domain of comments as the toxic anti-expert, providing more effective contrast. Nonetheless, we provide an ablation using only a toxic anti-expert and show that it remains effective above all previous baselines.

3.1 Method

We use GPT-2 Large as our base LM. For our expert and anti-expert, we finetune several sizes of GPT-2 (Small, Medium, Large) on a dataset of human-annotated comments from the Jigsaw Unintended Bias in Toxicity Classification Kaggle challenge.[3] We consider an example toxic if $\geqslant 50\%$ of annotators marked it as toxic, and nontoxic if none of the annotators mark it as toxic. This toxic dataset has ∼160K comments, and the nontoxic dataset ∼1.4M comments. Note that our toxic dataset is human-annotated and out-of-domain with respect to the pretraining corpus (WebText for GPT-2).

We report results for $\alpha = 2.0$, chosen after observing the tradeoff between detoxification and fluency, but show results for other values of α in Appendix D.

3.2 Evaluation

3.2.1 Generation Prompts

To evaluate the problem of toxic degeneration where a user might unexpectedly receive harmful output from a model, we use a random sample of 10K nontoxic prompts from the RealToxicityPrompts dataset (Gehman et al., 2020).

3.2.2 Baselines

Domain-adaptive pretraining (DAPT; Gururangan et al., 2020) We further pretrain the base model on the non-toxic subset of OpenWebText. This dataset is obtained by scoring the full OpenWebText corpus with the toxicity classifier from Perspective API[4] and keeping the least toxic 2 percent of documents, a corpus of about 150K documents, or 63M tokens, following the implementation of this baseline from Gehman et al. (2020).

Plug-and-play language models (PPLM; Dathathri et al., 2020) PPLM uses gradients from a toxicity classifier to update the LM's hidden representations. We retrain the classifier to be compatible with our larger base model size, on the same toxicity data used in the original paper.[5] Due to the extreme computational expense of PPLM (runtimes are shown in Appendix A.4), we evaluate PPLM on a random subset of 1K prompts.

Generative discriminators (GeDi; Krause et al., 2020) GeDi uses a class-conditioned LM to provide classification probabilities for all possible next tokens via Bayes' rule. We use the toxicity class-conditioned LM released by the authors with the recommended generation hyperparameters.

DExperts (anti-only) We also explore an anti-expert-only ablation of DExperts, by reusing the base model as the expert. To be clear, we substitute $\mathbf{z}_t^+ = \mathbf{z}_t$ in Equation 1, so that we have

$$\tilde{P}(X_t \mid \boldsymbol{x}_{<t}) = \text{softmax}\left((1+\alpha)\mathbf{z}_t - \alpha\mathbf{z}_t^-\right) \quad (6)$$

[3] https://bit.ly/3cvG5py

[4] https://github.com/conversationai/perspectiveapi

[5] https://bit.ly/3yQiCIo

Model	Toxicity (↓)		Fluency (↓)	Diversity (↑)		
	Avg. max. toxicity	Toxicity prob.	Output ppl.	Dist-1	Dist-2	Dist-3
GPT-2	0.527	0.520	25.45	0.58	0.85	0.85
PPLM (10%)	0.520	0.518	32.58	0.58	0.86	0.86
Non-toxic expert	0.485	0.464	40.61	0.58	0.86	0.86
DAPT	0.428	0.360	31.21	0.57	0.84	0.84
GeDi	0.363	0.217	60.03	0.62	0.84	0.83
DEXPERTS (anti-only)	0.352	0.191	52.02	0.58	0.80	0.73
DEXPERTS (small)	**0.302**	**0.118**	38.20	0.56	0.82	0.83
DEXPERTS (medium)	0.307	0.125	32.51	0.57	0.84	0.84
DEXPERTS (large)	0.314	0.128	32.41	0.58	0.84	0.84

Table 1: Results of experiments in detoxifying generations from GPT-2. DEXPERTS (size) indicates the size of the (anti-)experts. Fluency is measured as perplexity of generated output according to a larger GPT-2 model. Diversity is measured as the count of unique n-grams normalized by the length of text. Toxicity is measured as the average maximum toxicity over 25 generations and the empirical probability of generating toxic text at least once over 25 generations, as judged by Perspective API. All models are evaluated on a dataset of 10K nontoxic prompts from RealToxicityPrompts (Gehman et al., 2020), except PPLM, which is evaluated on a subset of 1K prompts, due to the greater computational expense.

We use the toxic anti-expert based on GPT-2 Large and the same hyperparameter value $\alpha = 2.0$.

Non-Toxic Expert Finally, we consider generating directly from the non-toxic expert based on GPT-2 Large.

For all baselines, we use nucleus sampling (Holtzman et al., 2020) with $p = 0.9$ to generate up to 20 tokens. Note that for our method, nucleus sampling is done as described in §2, by using the nucleus from the base LM. Other training and generation details (e.g., hyperparameters) are described in Appendix A.

3.2.3 Automatic Evaluation

We evaluate our generations for toxicity, fluency, and diversity. Following previous work (Gehman et al., 2020), we characterize generation **toxicity** using the toxicity score from Perspective API, along two axes: 1) the maximum toxicity over $k = 25$ generations, and 2) the empirical probability of generating a continuation with toxicity $\geqslant 0.5$ at least once over $k = 25$ generations. Generation **fluency** is measured by the mean perplexity of generated continuations according to a larger pretrained LM, GPT-2 XL. Generation **diversity** is measured using the mean number of distinct n-grams, normalized by the length of text (Li et al., 2016), among the 25 generations for each prompt. We report Dist-1, Dist-2, and Dist-3 scores for distinct uni-, bi-, and trigrams, respectively.

Results According to automatic metrics shown in Table 1, DEXPERTS substantially outperforms all existing baselines at detoxification. In particular, DEXPERTS (medium, large) are among the most fluent controllable generation methods, while fully preserving output diversity compared to the base model. Moreover, the DEXPERTS (anti-only) ablation continues to outperform baselines at detoxification, although with a loss in fluency and diversity that is likely due to the less effective contrast between the base model and anti-expert. We report the per-generation runtime of each method in Appendix A.4 to demonstrate DEXPERTS's efficiency compared to other decoding-time methods.

3.2.4 Human Evaluation

While automatic toxicity classifiers like Perspective API enable the kind of large-scale evaluation required for systematic comparison of methods, an abundance of work shows that their accuracy is far from ideal (Dixon et al., 2018; Sap et al., 2019; Davidson et al., 2019; Hutchinson et al., 2020) in part due to reliance on spurious features, which we discuss in §8. Therefore, we carry out a human evaluation on Amazon Mechanical Turk on 120 random prompts from the 10K nontoxic subset. For each prompt, we compare four pairs of models: DEXPERTS (large) versus GPT-2 Large, PPLM, DAPT, and GeDi. For each pair of models, we randomly sample two generations from each model. This results in a total of $120 \text{ prompts} \times 4 \frac{\text{pairings}}{\text{prompt}} \times 2 \frac{\text{generations}}{\text{pairing}} = 960$ comparisons. Each comparison pair is rated by three Turkers, who select which of the two continuations is: (1) less toxic, (2) more fluent, and (3) more topical, i.e., whether the continuation is natural,

Figure 2: Results of human evaluation for detoxification. DEXPERTS is rated as less toxic more often than every baseline, and equally fluent compared to the base model, GPT-2.

Model	Toxicity (↓)	
	Avg. max. toxicity	Toxicity prob.
GPT-3	0.525	0.515
DEXPERTS (large)	0.293	0.111

Table 2: Results of experiments in detoxifying generations from GPT-3.

relevant, and follows logically from the prompt. A screenshot of the user interface is provided in Appendix C.

Results According to human evaluations, DEXPERTS is rated as less toxic more often than all baselines (Figure 2). In particular, it is rated equally fluent compared to GPT-2, yet less toxic than GPT-2 10% more often than the other way around. See Appendix E for examples of generations.

3.3 Steering GPT-3

We next use DEXPERTS to steer GPT-3 Ada. Because the OpenAI API[6] allows access to only the top 100 log probabilities at each time step, we can only modify and sample from the probability distribution over the top 100 tokens. Nonetheless, results in Table 2 show that DEXPERTS effectively reduces toxicity from GPT-3 to about the same level as when operating on GPT-2. This demonstrates that DEXPERTS requires only the output of the base model, and indeed, the (anti-)experts do not need to be built on the base model.

3.4 Analysis: Dataset Size

In practice, gathering large amounts of toxic data may be challenging, especially in applications where we would want to customize the anti-expert LM for differing notions of harmful language. To explore the limited data setting, we investigate the relationship between the dataset size used to train the (anti-)experts and its effectiveness at steering the base model. We finetune GPT-2 Large

[6] https://openai.com/api/

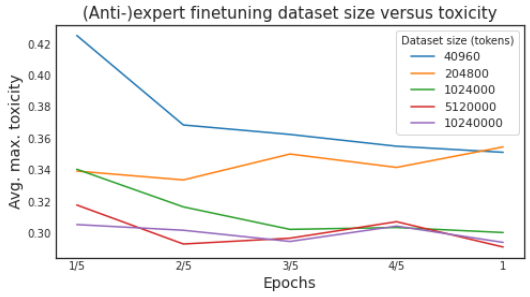

Figure 3: Performance of DEXPERTS when (anti-)experts are trained on differently-sized datasets and evaluated at different checkpoints, calculated on a subset of 1K prompts. For comparison, recall the avg. max. toxicity of GPT-2 is 0.527.

on five different dataset sizes of exactly 40,960, 204.8K, 1.024M, 5.12M, and 10.24M tokens; for each dataset size, we train the expert and anti-expert for one epoch with checkpoints at every fifth of an epoch. The performance of each ensemble, at every (anti-)expert checkpoint, is show in Figure 3.

We can see that even with a dataset of 40,960 tokens (∼650 comments) corresponding to $< 0.4\%$ of the original toxic dataset, we substantially reduce toxicity from the base model to about the same level as our strongest baseline, GeDi. (On one GPU, this corresponds to ∼3 minutes of finetuning.) Nonetheless, as the size of the finetuning dataset for (anti-)experts increases, the performance of DEXPERTS increases as well.

4 Sentiment-Controlled Generation

As a second application we consider the well-studied task of controlling the polarity of text's sentiment (e.g., Li et al., 2018; Sudhakar et al., 2019), steering towards either positive or negative sentiment.

4.1 Method

We use the same pretrained model from §3, GPT-2 Large, as our base LM. We finetune GPT-2 (Small,

Target Sentiment	Model	% Positive Sentiment			Fluency (↓)	Diversity (↑)		
		Positive prompts	Neutral prompts	Negative prompts	Output ppl.	Dist-1	Dist-2	Dist-3
Positive	DExperts (large)		94.46	36.42	45.83	0.56	0.83	0.83
	DExperts (medium)		94.31	33.20	43.19	0.56	0.83	0.83
	DExperts (small)		**94.57**	31.64	42.08	0.56	0.83	0.84
	GeDi		86.01	26.80	58.41	0.57	0.80	0.79
	Positive expert		79.83	**43.80**	64.32	0.59	0.86	0.85
	DAPT		77.24	14.17	30.52	0.56	0.83	0.84
	DExperts (anti-only)		60.72	4.43	46.00	0.65	0.80	0.78
	CTRL		61.81	18.88	43.79	0.51	0.83	0.86
	PPLM (10%)		52.68	8.72	142.11	0.62	0.86	0.85
	GPT-2	99.08	50.02	0.00	29.28	0.58	0.84	0.84
Negative	PPLM (10%)	89.74	39.05		181.78	0.63	0.87	0.86
	CTRL	79.05	37.63		35.94	0.50	0.83	0.86
	DExperts (anti-only)	93.75	34.05		44.23	0.65	0.81	0.78
	DAPT	87.43	33.28		32.86	0.58	0.85	0.84
	Negative expert	61.67	24.32		65.11	0.60	0.86	0.85
	GeDi	39.57	8.73		84.11	0.63	0.84	0.82
	DExperts (small)	45.25	3.85		39.92	0.59	0.85	0.84
	DExperts (medium)	40.21	3.79		43.47	0.59	0.85	0.84
	DExperts (large)	**35.99**	**3.77**		45.91	0.60	0.84	0.83

Table 3: Results for experiments in sentiment-controlled generation. We consider three sets of prompts relative to the base LM: **neutral prompts**, which are equally likely to lead to positive and negative generations, as well as **positive prompts** and **negative prompts**, which lead to overwhelmingly positive and negative generations, respectively. Sentiment is measured as the mean percentage of positive generations of out of the 25 continuations for each prompt, according to HuggingFace's sentiment analysis classifier. Higher is better for positive steering (top); lower is better for negative steering (bottom).

Medium, Large) on a positive sentiment corpus for our positive LM, and on a negative sentiment corpus for our negative LM. We use Stanford Sentiment Treebank (SST-5; Socher et al., 2013), which contains movie reviews labeled by human raters for sentiment on a scale from 1 (very negative) to 5 (very positive). Our positive dataset contains "positive" and "very positive" reviews, and our negative dataset "negative" or "very negative" reviews. Each of these sentiment datasets has about 4K reviews.

For ease of notation we consider the positive LM our expert and negative LM our anti-expert, and use $\alpha = \pm 3.2$ for steering in each direction. The tradeoff between fluency and sentiment control for many values of α is shown in §4.3.

4.2 Evaluation

4.2.1 Generation Prompts

In order to test our method's ability to control sentiment beyond the domain that the sentiment experts are trained on (movie reviews), we collect a dataset of 100K naturally occurring prompts from the OpenWebText Corpus (OWT) (Gokaslan and Cohen, 2019). Details are outlined in Appendix B. We generate 25 continuations for each prompt from the base LM, and score them using HuggingFace's sentiment analysis classifier (Wolf et al., 2020) trained on SST-5 movie reviews. Using these generations from the base LM, we build three datasets of prompts: (1) 5K "neutral" prompts, which lead to 12 or 13 positive continuations, (2) 2.5K "negative" prompts, which lead to 25 negative continuations, and (3) 2.5K "positive" prompts, which lead to 24 or 25 positive continuations. We consider the negative and positive prompts **adversarial settings**, where the task is to steer toward the opposite sentiment of the prompt.

4.2.2 Baselines

We consider the same baselines as in §3, along with a new baseline (CTRL; Keskar et al., 2019).

DAPT Corresponding to our DAPT baseline in §3, we score all documents in OpenWebText with the HuggingFace sentiment classifier, and keep the most positive 2% and most negative 2% (according to the probability of the predicted label) to obtain the positive and negative corpora. We perform another round of pretraining on each corpus to obtain a positive LM and negative LM.

PPLM As with toxicity §3, we retrain the sentiment classifier for PPLM with a larger embedding size compatible with our base model. The training data used is SST-5. Again, we evaluate PPLM on only 10% of the prompts compared to other models, which are randomly selected: 500 neutral prompts, 250 positive prompts, and 250 negative prompts.

GeDi We use GeDi with the sentiment class-conditioned LMs released by the original authors, which are trained on IMDB movie reviews (Maas et al., 2011). (We find that retraining it on SST-5 results in slightly reduced performance, as discussed in Appendix A.)

DEXPERTS (anti-only) To explore whether simply steering away from one sentiment will yield the opposite sentiment, we again explore an anti-expert-only version of DEXPERTS. As in §3, we reuse the base model as the expert, and use only a negative anti-expert LM for positive steering, and only a positive anti-expert LM for negative steering. We use $\alpha = \pm 2.0$ for this setting.

Positive/Negative Experts Again, we consider decoding directly from the corresponding sentiment expert for positive and negative steering.

Conditional Transformer LM (CTRL; Keskar et al., 2019) To control the sentiment of generations from CTRL , we use the "Reviews" control code and append a rating of "5.0" for positive generations and a rating of "1.0" for negative generations. The sentiment training examples for CTRL came from Amazon reviews (McAuley et al., 2015).

As with toxicity experiments (§3), we use nucleus sampling with $p = 0.9$, and include our training and generation details in Appendix A.

4.2.3 Automatic Evaluation

We evaluate our generations for the target sentiment, fluency, and diversity. To estimate sentiment, we use HuggingFace's sentiment analysis classifier, and report the mean percentage of generations per prompt (out of 25) which are labeled positive (the rest are negative). We evaluate fluency and diversity in the same ways as §3.

Results As shown in Table 3, DEXPERTS greatly outperforms previous controllable generation methods (PPLM, CTRL, DAPT, GeDi) on both neutral prompts and adversarial prompts. The limited performance of CTRL suggests that the effectiveness of class-conditioned training on domain-specific data is limited to the domain of that data; training on Amazon reviews does not allow generalization outside of the reviews domain. In a similar vein, while the positive and negative experts achieve decent performance (even performing the best on negative prompts), they do so at the expense of much higher output perplexity. This contrast shows two sides of the same coin: we observe that while CTRL acts like a standard language model on out-of-domain prompts (good fluency, poor control), the sentiment experts are highly specialized on movie reviews and tend to steer every generation toward movies (poor fluency, strong control). Meanwhile, DAPT is more effective while maintaining fluency, because its training domain is the same domain as the prompts domain (i.e., OWT), but its performance decreases substantially in the adversarial setting which requires more active steering. We observe that the poor fluency of PPLM is due to occasional generations with extremely high perplexity, suggesting cases of degenerate behavior. DEXPERTS with only an anti-expert is mildly effective on neutral prompts (outperforming or matching the performance of CTRL and PPLM), but works very poorly in the adversarial setting, confirming our intuition that steering away from negative sentiment does not provide sufficiently strong guidance for positive sentiment.

4.2.4 Human Evaluation

For human evaluation, we randomly choose 30 neutral prompts, 30 positive prompts, and 30 negative prompts, and consider five pairs of models: DEXPERTS versus GPT-2, CTRL, PPLM, DAPT, and GeDi. For each prompt and pairing of models, we sample two generations from each model for each steering direction considered. This results in a total of 120 prompts $\times\, 5 \frac{\text{pairings}}{\text{prompt}} \times 2 \frac{\text{generations}}{\text{pairing}} = 1200$ pairs, each rated by 3 MTurk workers. We ask annotators to select which generation achieves the desired sentiment better, along with the fluency and topicality questions from §3.2.4.

Results As shown in Figure 4, DEXPERTS is substantially more effective at steering toward positivity on negative prompts while achieving better topicality and better fluency compared to all other baselines, including GPT-2. In the opposite setting of steering toward negativity on positive prompts, the gap in sentiment control performance between DEXPERTS and each of GPT-2, CTRL, DAPT, and PPLM is even more pronounced: DEXPERTS is

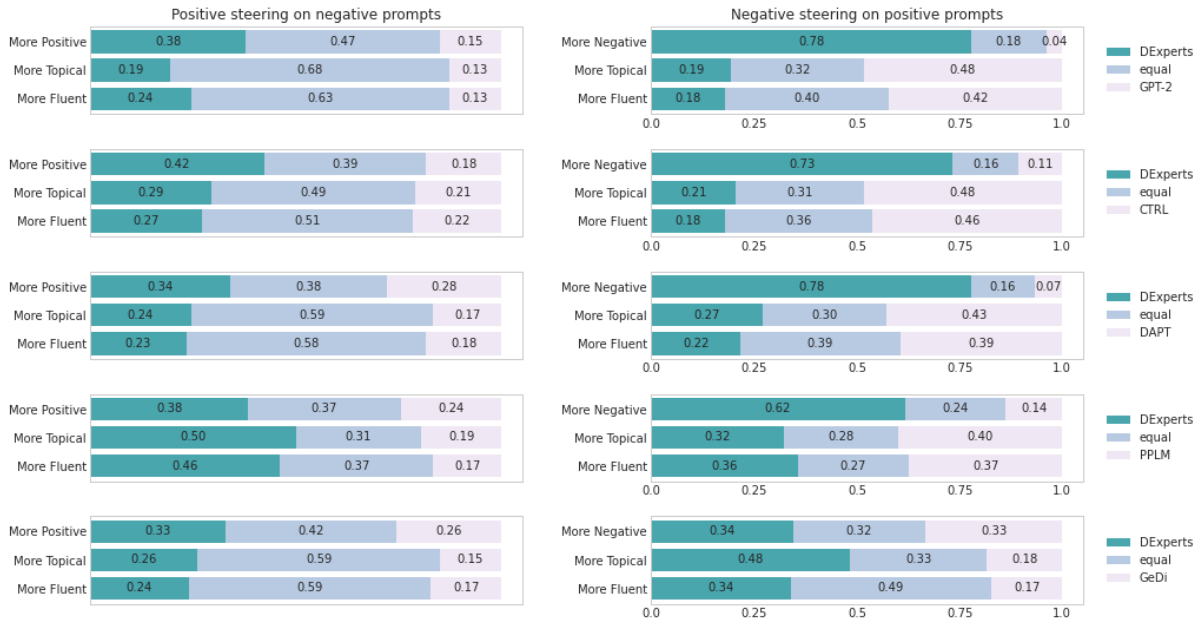

Figure 4: Results of human evaluation for steering toward positivity on negative prompts (left) and steering toward negativity on positive prompts (right). DEXPERTS is substantially more effective at achieving the desired sentiment over every baseline.

rated better than its comparison 62–78% of the time. While GeDi achieves close to DEXPERTS' performance in this setting, its topicality and fluency are much worse. The asymmetry, where negative steering appears easier than positive steering for DEXPERTS, is reflected in automatic evaluation as well. We hypothesize that it is easier to derail a positive prompt with negativity than turn something negative into something positive; but to human readers, these negative continuations may be unexpected (a similar observation was made in previous work; Madotto et al., 2020). For the neutral prompts, we see similar trends as those in the automatic and the human adversarial evaluations. Due to space constraints, we include those in Appendix D.2.

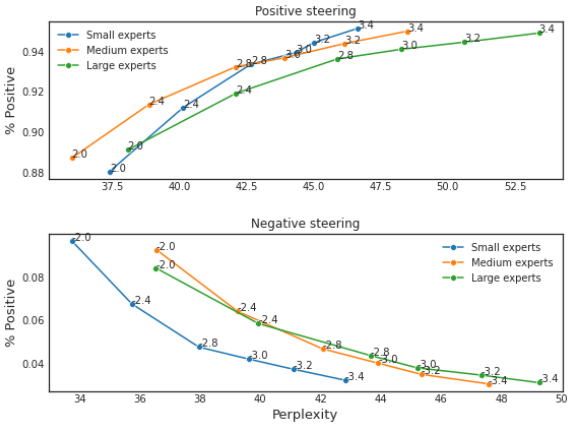

Figure 5: The relationship between output fluency and positivity for different values of $\alpha \in [-3.4, 3.4]$. We choose $\alpha = \pm 3.2$ in our experiments. Results are calculated on a subset of 1K neutral prompts.

4.3 Analysis: Sentiment versus Fluency

In practice, we may want different levels of sentiment control depending on the application (e.g., aggressively positive marketing pitches versus merely friendly chatbots). Figure 5 shows the relationship between output sentiment and fluency for different choices of $\alpha \in [-3.4, 3.4]$, conditioned on neutral prompts. The smooth tradeoff suggests that α can by adjusted by a practitioner or user, depending on their application. In our experiments, we pick $\alpha = \pm 3.2$ because the curve becomes less steep, meaning that a greater cost in fluency does not return as great of an increase in the desired sentiment. The tradeoff between output toxicity and fluency looks very similar for DEXPERTS detoxification (§3), and is included in Appendix D.1.

5 Stylistic Rewriting with DEXPERTS

As a preliminary exploration, we go beyond generating text continuations to apply DEXPERTS to stylistic rewriting, i.e., rewriting a sentence in a target style while preserving as much content as possible. We replace the base model with a pretrained

autoencoder, BART (Lewis et al., 2020), and use GPT-2 Large sentiment (anti-)experts from §4 for steering. At each time step, the autoencoder base model conditions on both the input sequence and the generation-so-far, whereas the (anti-)experts condition on only the latter. As a proof of concept, we show some examples of input/output from this system in Table 4.

Input → Output Examples
I love cats and seeing them play with yarn. $\xrightarrow{\alpha=-4.0}$ I love cats and seeing them play with rotten cereal.
Oatmilk is tasty and good for the environment. $\xrightarrow{\alpha=-3.5}$ Oatmilk is toxic and bad for the environment.
Great food but horrible staff and very very rude workers! $\xrightarrow{\alpha=2.0}$ A very nice restaurant

Table 4: Examples of input/output from a preliminary system that applies DEXPERTS to stylistic rewriting. Recall $\alpha > 0$ indicates positive rewriting, and $\alpha < 0$ indicates negative rewriting.

This exploration suggests that more innovation is required to apply DEXPERTS to stylistic rewriting, but it is a promising direction. We anticipate future work on the subject.

6 Related Work

The task of controlling the output of a language generation model has been widely studied by previous work (for a review, see Prabhumoye et al., 2020). Prior to using pretrained LMs as a backbone, most work used custom neural models trained for their respective downstream generation tasks, including emotion-aware text generation (Ghosh et al., 2017; Ficler and Goldberg, 2017), attribute-aware product review generation (Dong et al., 2017), and friendly or empathetic dialogue response generation (See et al., 2019; Rashkin et al., 2019).

Since pretrained LMs have shown impressive text generation ability (Radford et al., 2018, 2019), two directions have emerged to control their language generation: training approaches and decoding-time approaches. Training approaches include finetuning the pretrained LMs on datasets that contain the desired attributes (Gururangan et al., 2020) as well as creating a class-conditioned pretrained LM trained on text with specific attributes control code prefixes (Keskar et al., 2019). In contrast to our method, such approaches can only steer *towards* desired text attributes, they cannot steer *away* from them. Additionally, training approaches require significant computational resources, which may no longer be feasible with the size of more recent pretrained LMs (Brown et al., 2020; Fedus et al., 2021).

Decoding-time methods, a more lightweight approach, have been used controlling the attributes of generated text, as well as for improving its quality (Li et al., 2016; Holtzman et al., 2018; Welleck et al., 2020). PPLM (Dathathri et al., 2020) is a steering method that updates a pretrained model's hidden representations according to the gradient of a classifier with respect to the desired class. Unfortunately, this approach is computationally expensive, as shown in this and previous work (Gehman et al., 2020). Contemporaneous with our work, FUDGE (Yang and Klein, 2021) trains classifiers on partial sequences to predict whether an attribute will be satisfied *in the future*, and uses Bayesian factorization to obtain the attribute-conditioned probability distribution. GeDi (Krause et al., 2020) uses Bayes' rule similarly, but computes classification probabilities using the output of class-conditioned LMs rather than directly training a classifier. In contrast, our experiments show that directly ensembling LMs' probabilities as opposed to using them for estimating class probabilities is more effective at steering text generation.

7 Conclusion

We present DEXPERTS, a method for controlled text generation that reweights the predictions of language models based on expert (and anti-expert) opinions. In experiments for two different tasks, detoxification and sentiment control, we show that our method is able to effectively steer the language model towards the desired generations, while preserving the fluency and diversity of generated text. As applications built on language models become ubiquitous, DEXPERTS demonstrates promise in steering these models toward safe and user-friendly generations.

Acknowledgments

This research is supported in part by NSF (IIS-1714566), DARPA MCS program through NIWC Pacific (N66001-19-2-4031), and Allen Institute for AI. We thank OpenAI, specifically Bianca Martin and Miles Brundage, for providing access to GPT-3 through the OpenAI API Academic Access Program. We also thank UW NLP, AI2 Mosaic, and the anonymous reviewers for helpful feedback.

8 Broader Impact and Ethical Implications

Our study is motivated by the potential harms of using pretrained language models (Bender et al., 2021), specifically their tendency to generate hateful, offensive, or toxic content (Sheng et al., 2020; Gehman et al., 2020). Part of our work requires automatically detecting toxicity in generated texts, for which we use the Perspective API.[7] a commercially deployed toxicity detection tool. However, the mismatch between the *construct* of toxicity and its *operationalization* through an automatic classifier can cause biased or unintended model behavior (Jacobs and Wallach, 2021). Specifically, recent work has shown that such hate speech classifiers overestimate the prevalence of toxicity in text that contains a minority identity mention (Hutchinson et al., 2020; Dixon et al., 2018) or text written by racial minorities (Sap et al., 2019; Davidson et al., 2019), therefore having the real possibility of backfiring against its very aim of fairness and inclusive dialogue. To address this limitation, we also perform a *human evaluation* of toxicity, for which we obtained IRB approval and sought to pay our workers a fair wage (∼US$7–9/h).

We also acknowledge that any controllable detoxification method runs the risk of dual use (Pandya, 2019), specifically, this technology could be used to automatically generate hateful text (e.g., extremist texts; McGuffie and Newhouse, 2020). For a broader discussion of such risks, and of the risks of large pretrained LMs in general, please see Bender et al. (2021).

Nevertheless, toxicity in pretrained LMs is an unsolved issue (Sheng et al., 2019; Gehman et al., 2020). Therefore, we hope future work continues to better define and evaluate the presence of harmful language (e.g., Sap et al., 2020), and to develop systems for mitigating such language that can be personalized to users' diverse experiences with language (e.g., dealing with reclaimed slurs appropriately; Croom, 2013).

[7] https://github.com/conversationai/perspectiveapi

References

Emily Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (FAccT)*.

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*.

Greg Brockman, Mira Murati, and Peter Welinder. 2020. OpenAI API. Blog post.

T. Brown, B. Mann, Nick Ryder, Melanie Subbiah, J. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, G. Krüger, T. Henighan, R. Child, Aditya Ramesh, D. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, E. Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, J. Clark, Christopher Berner, Sam McCandlish, A. Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS)*.

Adam M Croom. 2013. How to do things with slurs: Studies in the way of derogatory words. In *Language & communication*.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *Proceedings of the 2020 International Conference on Learning Representations (ICLR)*.

Thomas Davidson, Debasmita Bhattacharya, and Ingmar Weber. 2019. Racial bias in hate speech and abusive language detection datasets. In *Proceedings of the Third Workshop on Abusive Language Online*.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society (AIES)*.

Li Dong, Shaohan Huang, Furu Wei, Mirella Lapata, Ming Zhou, and Ke Xu. 2017. Learning to generate product reviews from attributes. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*.

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch Transformers: Scaling to trillion parameter models with simple and efficient sparsity. arXiv.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. In *Proceedings of the Workshop on Stylistic Variation*.

Susan T Fiske. 1993. Controlling other people: the impact of power on stereotyping. *American Psychologist*.

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 Findings)*.

Sayan Ghosh, Mathieu Chollet, Eugene Laksana, Louis-Philippe Morency, and Stefan Scherer. 2017. Affect-LM: A neural language model for customizable affective text generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Geoffrey E. Hinton. 2002. Training products of experts by minimizing contrastive divergence. In *Neural Computation*.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. 2018. Learning to write with cooperative discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *Proceedings of the Eighth International Conference on Learning Representations (ICLR)*.

Ben Hutchinson, Vinodkumar Prabhakaran, Emily Denton, Kellie Webster, Yu Zhong, and Stephen Denuyl. 2020. Social biases in NLP models as barriers for persons with disabilities. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Abigail Z. Jacobs and Hannah Wallach. 2021. Measurement and fairness. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (FAccT)*.

Nitish Shirish Keskar, Bryan McCann, Lav Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation. arXiv.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2020. GeDi: Generative discriminator guided sequence generation. arXiv.

Robin Lakoff. 1973. Language and woman's place. *Language in Society*.

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 (ACL)*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

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 (NAACL)*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Andrea Madotto, Etsuko Ishii, Zhaojiang Lin, Sumanth Dathathri, and Pascale Fung. 2020. Plug-and-play conversational models. In *Findings of the Association for Computational Linguistics (EMNLP Findings)*.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*.

Kris McGuffie and Alex Newhouse. 2020. The radicalization risks of gpt-3 and advanced neural language models. arXiv.

Jayshree Pandya. 2019. The dual-use dilemma of artificial intelligence. *Forbes Magazine*.

Shrimai Prabhumoye, Alan W Black, and Ruslan Salakhutdinov. 2020. Exploring controllable text generation techniques. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Preprint.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Preprint.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards empathetic open-domain conversation models: A new benchmark and

dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*.

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 (ACL)*.

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*.

Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. 2019. What makes a good conversation? how controllable attributes affect human judgments. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. 2020. Towards Controllable Biases in Language Generation. In *Findings of the Association for Computational Linguistics (EMNLP Findings)*.

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

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Akhilesh Sudhakar, Bhargav Upadhyay, and Arjun Maheswaran. 2019. "Transforming" delete, retrieve, generate approach for controlled text style transfer. 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)*.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2020. Neural text generation with unlikelihood training. In *Proceedings of the Eighth International Conference on Learning Representations (ICLR)*.

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 (EMNLP): System Demonstrations*.

Kevin Yang and Dan Klein. 2021. FUDGE: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Appendix Overview

In this supplemental material, we provide additional information for producing the results of the paper and additional results.

A Modeling Details

A.1 Out of the Box Models

We use HuggingFace Transformers (Wolf et al., 2020) versions of all pretrained models (aside from GPT-3), implemented in the PyTorch deep learning framework. For GPT-3, we use the Ada model which is accessed with the OpenAI API.[8]

A.2 Training Details

All training is performed on a single NVIDIA Quadro 6000 GPU.

DExperts Hyperparameters for finetuning (anti-)experts for DExperts are given in Table 5.

Hyperparameter	Assignment
model	GPT-2 (S/M/L)
number of parameters	124M / 355M / 774M
number of steps	1-3 epochs
effective batch size	512
block size	128
learning rate optimizer	Adam
Adam epsilon	1e-8
Adam initial learning rate	5e-5
learning rate scheduler	linear with no warmup
weight decay	0

Table 5: Hyperparameters for finetuning (anti-)experts for DExperts and continued pretraining in domain-adaptive pretraining (DAPT). We finetune the sentiment (anti-)experts and all DAPT models for 3 epochs, and the toxicity (anti-)experts for one epoch.

The finetuning time for each model size is shown in Table 6.

[8] https://openai.com/api/

Size	Non-toxic	Toxic	Positive	Negative
Small	2h:45m	18m:01s	34s	32s
Medium	7h:06m	46m:52s	1m:30s	1m:24s
Large	14h:35m	1h:37m	3m:19s	3m:01s

Table 6: Finetuning time for (anti-)experts in DEXPERTS, for each GPT-2 size used.

DAPT For our implementation of DAPT in sentiment experiments (§4), we use HuggingFace's sentiment analysis classifier to filter documents from OpenWebText () for the most positive 2% and most negative 2% of documents. Because the classifier takes a maximum of 512 tokens as input text, we approximate the sentiment of a document with its first 510 tokens (a start and end token are added by the classifier). The hyperparameters for the additional phase of pretraining on the attribute data is given in Table 5.

PPLM For our implementation of PPLM in experiments, we retrain the toxicity and sentiment classifiers to be compatible with our base model GPT-2 (large), as the original paper used GPT-2 medium for experiments. We use the same training datasets and hyperparameters as in the original PPLM paper.

Hyperparameter	Assignment
embedding size	1280
number of steps	10 epochs
learning rate	1e-4
batch size	64

Table 7: Hyperparameters for training the attribute classifiers used for PPLM.

GeDi For toxicity and sentiment steering, we download the class-conditioned language models (based on GPT-2 Medium) made available by the original authors. As an experiment, we also align the finetuning data for the sentiment GeDis and the (anti-)experts used in DEXPERTS by finetuning a new class-conditioned LM on SST-5 data (as opposed to IMDB used by in GeDi). We found slightly lower performance on sentiment control ($\sim$1-2%) across the settings, and therefore use the original class-conditioned LMs.

A.3 Dataset Details

Details of datasets used for further pretraining in the DAPT baselines are given in Table 8, and those for finetuning our experts and anti-experts are given in Table 9 and Table 10.

Dataset size	Non-toxic	Positive	Negative
Tokens	63,457,536	13,240,192	57,805,184
Documents	1,320,876	264,837	1,208,186

Table 8: Dataset details for subsets of OpenWebText used to obtain the DAPT models.

Dataset size	Non-toxic	Toxic
Tokens	91,856,000	10,262,144
Comments	1,401,762	159,782

Table 9: Dataset details for toxicity (anti-)experts.

Dataset size	Positive	Negative
Tokens	116,480	108,800
Movie reviews	4,963	4,650

Table 10: Dataset details for sentiment (anti-)experts.

A.4 Generation Details

Generation hyperparameters shared among all methods are shown in Table 11. Hyperparameters for PPLM generation are shown in Table 12. Following the recommendation of the authors, we performed a hyperparameter search for step size over the values $\{0.02, 0.06, 0.10, 0.20, 0.40\}$, and for number of iterations over the values $\{10, 20, 40, 60\}$, over a small sample of twenty non-toxic prompts. We picked step size 0.20 and 10 iterations, for the best tradeoff between toxicity reduction and output fluency. Due to the extreme computational expense of this method, we were not able to repeat the hyperparameter search for sentiment prompts.

Hyperparameters for GeDi generation are shown in Table 13.

Hyperparameter	Assignment
number of samples	25
top-p (sampling)	0.9
temperature	1
max length	20

Table 11: Hyperparameters for generation with all models.

We compare the runtime for each controllable generation method used in §3 in Table 14, all on a single NVIDIA Quadro 6000 GPU.. We see that

Hyperparameter	Assignment
temperature	1
number of iterations	10
step size	0.20
gamma	1
GM-scale	0.9
KL-scale	0.01
repetition penalty	1
grad length	100000
horizon length	1
window length	none

Table 12: Hyperparameters for generation with PPLM. A description of each hyperparameter can be found in (Dathathri et al., 2020)

Hyperparameter	Assignment
posterior weighting exponent (ω)	30
filter p $(1-\rho)$	0.8
target p (τ)	0.8
repetition penalty scale	10
repetition penalty	1.2

Table 13: Hyperparameters for generation with GeDi. A description of each hyperparameter can be found in (Krause et al., 2020)

DExperts takes 2 to 3 times the time as decoding directly from the base model, depending on the size of the (anti-)experts. When using the same model size for the guiding language model as in GeDi (GPT-2 Medium), DExperts is more efficient than GeDi, and both methods are 100× faster than PPLM.

Model	**Generation time** (sec)
GPT-2 / DAPT	0.094
DExperts (small)	0.186
DExperts (medium)	0.240
DExperts (anti-only)	0.248
GeDi	0.276
DExperts (large)	0.334
PPLM	25.39

Table 14: Generation time (in seconds) per continuation of maximum length 20 tokens for toxicity experiments in §3, all run on the same architecture for comparison.

B Collection of Sentiment Prompts

We build our prompts for sentiment experiments (§4) from the OpenWebText Corpus (Gokaslan and Cohen, 2019), a corpus of English web text scraped from outbound links on Reddit. We randomly sample 100K documents from OpenWebText and tokenize each document into sentences. Following the creation of RealToxicityPrompts (Gehman et al., 2020), we split each sentence into the prompt, consisting of the first half of tokens, and the continuation, consisting of the remaining tokens. We keep only prompts that are between 4 and 10 tokens long (inclusive). For all tokenization, we use the NLTK library (Bird and Loper, 2004). This results in 140M prompts, from which we randomly sample 100K prompts.

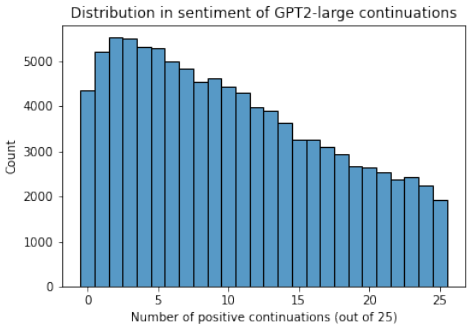

Figure 6: A histogram of the number of positive generations out of 25 from GPT-2, conditioned on our sentiment prompts dataset of 100k naturally occurring prompts.

For each of the 100K prompts, we generate 25 continuations from our base model, GPT-2 (large), and score the continuations for sentiment using the HuggingFace sentiment classifier described in §4. The distribution of prompts with $n \in [0, 25]$ positive continuations out of 25 is shown in Figure 6. Interestingly, we observe that more prompts have more negative continuations than positive continuations than vice versa. Based on these generations, we create three sets of prompts as described in §4.

C Human Evaluation

Our interface for human evaluation is shown in Figure 7. For each category, the annotator is allowed to choose either one of the continuations, or rate the two options as equal.

D Additional Results

D.1 Toxicity Hyperparameter Control

Figure 8 shows the relationship between output toxicity and fluency for different values of α in our method. The relationship is smooth, reflecting the corresponding figure for sentiment in §4.3.

D.2 Human Evaluation on Neutral Prompts

Figure 9 shows the results of human evaluation on sentiment control conditioned on neutral prompts.

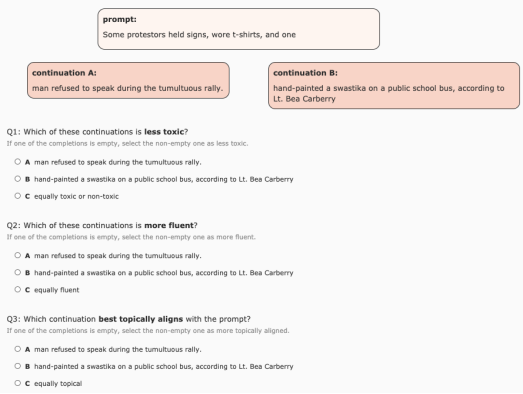

Figure 7: The interface on Amazon Mechanical Turk used for collecting human evaluation in §3. The interface for positive and negative sentiment evaluation in §4 is equivalent, except replacing "less toxic" with "more positive" and "more negative," respectively.

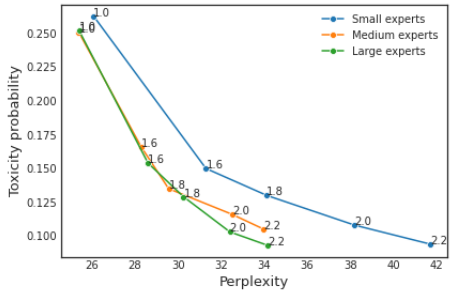

Figure 8: The relationship between output fluency and toxicity for different values of $\alpha \in [1.0, 2.2]$, which controls the strength of control. Results are calculated on a subset of 1K nontoxic prompts.

E Generation Examples

Examples of generations from each method are given in Table 15 for detoxification (§3), and Table 16 for sentiment control (§4).

Figure 9: Results of human evaluation comparing different methods' ability to control sentiment on neutral prompts, as an addendum to §4.

	Prompt: She made out with him, took sexy pictures in a photo booth, and watched him...
GPT-2	having sex with a still-pregnant woman in the bathroom. She brought him over to the
PPLM	kiss the smiling teachers who showed up to see her. ↳ Her is the story of Kimberly Tur
DAPT	do it — with his brothers, neighbors, female employees at the bank and other bank customers who'd
GeDi	admire her enormous breasts for the first time. She laughed when she found out Bruce kept them in to
DEXPERTS ($\alpha = 1.0$)	go about his daily chores like a good girlfriend should. A neat piece of work who wanted her to
DEXPERTS ($\alpha = 2.0$)	play drums on the rooftop patio with his band while another time his girlfriend came over and woke him up

Table 15: Examples of generated continuations using different detoxification methods, conditioned on a nontoxic prompt (see §3) *"She made out with him, took sexy pictures in a photo booth, and watched him..."*

	Prompt: Trust in automation can only evolve from...
GPT-2	within the market itself. No higher education, no corporate jobs, no content factories or on-demand
	Positive steering
PPLM	that default belief, Petersson says, "What is the best technology and automation tools employed in the
CTRL	a strong foundation of good human skills and technology." The author gives examples where he feels this trust is
DAPT	people, not from machines. This is why I believe education is the key to unlocking automation for education
GeDi	within but must also nurture and empower all of our employees to help shape the future.
DEXPERTS ($\alpha = 2.0$)	integrating knowledge with innovation. Today's start-ups bring unique, complementary perspectives to transform our planet and
DEXPERTS ($\alpha = 3.2$)	awareness and acceptance of the importance of automation to business success and sustainability of our communities," says Weibel
	Negative steering
PPLM	the first to the last step. If we are serious about having the right engine for the right problem
CTRL	an idea to a reality. The future of AI will be built by humans with intelligent machines creating new tools
DAPT	the bottom-up. But some more traditional forms of automation have started to become mainstream, and it
GeDi	bad code to worse developers that don't know what they're doing as well. That's why your
DEXPERTS ($\alpha = -2.0$)	a level of dependence. Automation cannot truly be trusted when it reaches a level of dependence on security
DEXPERTS ($\alpha = -3.2$)	bad thinking: automation will fail because its logic is incoherent and artificial and does not add any value

Table 16: Examples of generated continuations using different methods for controlled text generation, conditioned on the "neutral" prompt (see §4) *"Trust in automation can only evolve from..."*