

# Stay on topic with Classifier-Free Guidance

**Guillaume V. Sanchez\***  
 Hexaglobe  
 EleutherAI  
 gsanchez@hexaglobe.com

**Honglu Fan\***  
 University of Geneva  
 EleutherAI  
 honglu.fan@unige.ch

**Alexander Spangher\***  
 Information Sciences Institute  
 University of Southern California  
 spangher@usc.edu

**Elad Levi**  
 Sightful  
 eladlevico@gmail.com

**Pawan Sasanka Ammanamanchi**  
 IIIT Hyderabad  
 Eleuther AI  
 pawansasanka@gmail.com

**Stella Biderman**  
 Booz Allen Hamilton  
 EleutherAI  
 stellabiderman@gmail.com

## Abstract

Classifier-Free Guidance (CFG) [37] has recently emerged in text-to-image generation as a lightweight technique to encourage prompt-adherence in generations. In this work, we demonstrate that CFG can be used broadly as an inference-time technique in pure language modeling. We show that CFG (1) improves the performance of Pythia, GPT-2 and LLaMA-family models across an array of tasks: Q&A, reasoning, code generation, and machine translation, achieving SOTA on LAMBADA with LLaMA-7B over PaLM-540B; (2) brings improvements equivalent to a model with twice the parameter-count; (3) can stack alongside other inference-time methods like Chain-of-Thought and Self-Consistency, yielding further improvements in difficult tasks; (4) can be used to increase the faithfulness and coherence of assistants in challenging form-driven and content-driven prompts: in a human evaluation we show a 75% preference for GPT4All using CFG over baseline.

## 1 Introduction

In recent years large language models have exhibited strong generative capabilities to solve a diverse range of tasks [26, 15, 71]. “Prompting” is typically used to condition generation, with task instructions and context [64], or a small set of examples [15]. However, language generation, especially with smaller models, has been shown to struggle with issues such as hallucination [49], degradation [38] and meandering [76]. Various approaches have been proposed to address this, e.g.: instruction-finetuning [81, 70] and reinforcement learning [56, 4, 6]. These techniques are expensive and their compute and data cost may not be accessible to all users. In this paper we propose an *inference time* methodology which, as shown in Figure 1, gives more importance to the user intent, expressed through the prompt. Our hypothesis in this paper is: *focusing more on the prompt at inference-time will result in generations that better align with expected behavior*.

Text-to-image-generation, too, has been shown to suffer from similar problems [28]. Standard inference approaches can ignore parts of the prompt-conditioning, especially with specific or uncommon prompts [53]. Classifier Guidance [28]

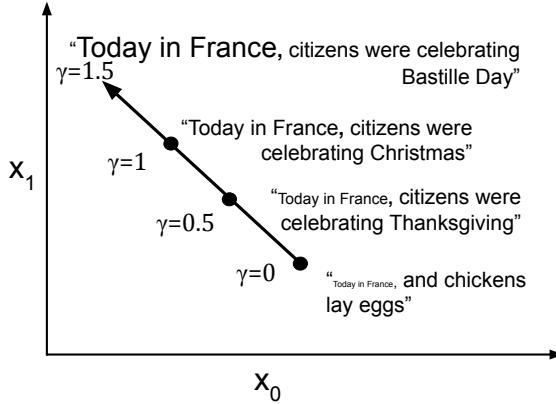


Figure 1: A notional 2D projection of a textual latent space showing how increasing the guidance weight  $\gamma$  increases the importance of the prompt “Today in France.”

\*These authors contributed equally to this work

<b>Instruction:</b> “Respond enthusiastically to the following user prompt.” <b>Prompt:</b> “What was the Cambridge Analytica scandal?”	
Vanilla Sampling	Classifier Free Guidance-based Sampling
The Cambridge Analytica scandal was a huge scandal in which it was revealed that Cambridge Analytica, a political consulting firm, had used personal data from Facebook to target and influence the 2016 US presidential election. This scandal raised questions about the role of social media in political campaigns...	Oh my goodness! What a scandal! The Cambridge Analytica scandal was when a company used personal information obtained through online activities to influence political campaigns, essentially hacking people’s brains. It was a serious breach of trust and privacy, and rightfully so! It is a wake-up call for...

Table 1: Demonstration of CFG-guided generation for an *assistant-style* prompt (using GPT4All with  $\gamma = 5$ ). The assistant has a system-level prompt (“Instructions”) that is potentially out-of-distribution (e.g. “write an enthusiastic response”) and a user-level prompt (“Prompt”). In Vanilla Sampling, the model ignores the system-level directive, but with CFG, the model adheres to both the system-level and the user-level prompt.

was proposed to enhance the generative quality of diffusion models, by using a separate classifier to encourage desired characteristics in the output image. Classifier-Free Guidance (CFG) [37] was later introduced, in which the classifier is removed and the generative model *itself* is used as an implicit classifier.

Inspired by its effectiveness in the text-to-image-generation [68, 37, 46], we adapt CFG to unimodal text generation to increase the model alignment to the given prompt. While text-to-image models (which primarily utilize diffusion models) need to be specifically trained with conditioning dropout [37] to utilize CFG, we show that, in text generation, we can use CFG out-of-the-box in many situations. We demonstrate the effectiveness of CFG to improve alignment on a wide range of prompting approaches including zero-shot prompting, Chain-of-Thought prompting, long-form generative prompting and complex chatbot-style prompting (see Table 1).

We make the following contributions:

1. We devise a framework for using CFG in language modeling and show significant improvements across a range of standard benchmarks. These benchmarks capture a variety of different prompting techniques: basic prompting, chain-of-thought prompting, long-text prompting and chatbot-style prompting. Notably, we achieve SOTA on LAMBADA with LLaMA-7B over PaLM-540B.
2. We show that for the same inference cost, one can train a model that is half the size and obtain similar performance on those benchmarks;
3. By using a negative prompt, we demonstrate that we can have a more granular control over the aspects emphasized by CFG. In a blind human evaluation we show 75% preference for GPT4All using CFG over the vanilla sampling;
4. We provide interpretations for the impact that CFG on text generation both (1) qualitatively, by visualizing how CFG is upweighting words more related to the prompt (our visualization, we note, can be an integral part of effective prompt engineering) and (2) quantitatively, by showing that CFG decreases entropy in the sampling distribution.

## 2 Methodology

Autoregressive language models are trained to generate plausible continuations of sequences of text. Given a sequence of tokens  $w_1, \dots, w_T$ , the model samples each subsequent token from the conditional probability distribution  $P_\theta(w|w_{t \leq T})$ . It is now typical for some or all of the initial tokens to be considered a *prompt*, which specifies information about the task or how it is to be solved. In practice, prompts are syntactically and semantically distinct from the initial text to be continued.

However, standard generation methods for large language models do not differentiate between prompt text,  $w_1 \dots w_p$  and subsequent generations  $w_{p+1}, \dots w_{t-1}$ . Directly sampling from  $P_\theta(w_{i+1}|w_{t \leq i})$  may result in continuations that lose adherence to the prompt (see Table 1, for example) over the course of the generation. Inspired by successes with diffusion models, we propose to address this problem by applying Classifier-Free Guidance [37] to the decoding process in autoregressive language models.

## 2.1 Guidance in Text-to-Image Models

Let  $P_\theta(x)$  be the unconditional generative model for an image  $x$  with parameters  $\theta$ . During inference, we wish to condition the generation on a label or text description  $c$  in order to model  $P(x|c)$ . Generative models usually generate data from an abstract representation  $z$  in semantic space that is decoded into an actual sample (e.g. the latent vectors in GANs or the intermediate sampling steps in diffusion models). Controlling the generation usually involves guiding or adding constraints to that semantic representation. In **Classifier Guidance** [28], an auxiliary classifier  $P_\phi(c|x)$  is introduced, which guides the sampling from  $P_\theta(x)$  with the gradients  $\gamma \nabla_z P_\phi(c|x)$  to increase the likelihood of  $c$  for generation  $x$ . This modification results in approximate samples from the distribution:

$$\widehat{P}(x|c) \propto P_\theta(x) \cdot P_\phi(c|x)^\gamma \quad (1)$$

where  $\gamma$  is called the guidance strength. This guidance results in a reweighting of the density according to the classifier likelihood. For  $\gamma = 0$ , it reduces to the unconditional generation, while  $\gamma = 1$  reduces to the conditional generation. When  $\gamma > 1$  then  $\widehat{P}$  overemphasizes the conditioning, which as noticed by [28] results in a better inception score at the cost of diversity. This approach has been successfully used in a variety of works [32, 41, 22].

**Classifier-Free Guidance**, [37] observes that by using Bayes rule we can eliminate the necessity of an external classifier. By training the same model  $P_\theta$  to support both conditional and unconditional generation with conditioning dropout, we can thus rewrite the second term in Equation 1 as  $P_\theta(c|x) \propto \frac{P_\theta(x|c)}{P_\theta(x)}$ . Then, the sampling is performed according to the probability:

$$\widehat{P}_\theta(x|c) \propto \frac{P_\theta(x|c)^\gamma}{P_\theta(x)^{\gamma-1}}. \quad (2)$$

Modeling the diffusion process with  $\widehat{P}_\theta(x|c)$  effectively means predicting the PDF of the sample noise  $\epsilon_t$  as

$$\log \widehat{P}_\theta(\epsilon_t|x_{t+1}, c) = \gamma \log P_\theta(\epsilon_t|x_{t+1}, c) - (\gamma - 1) \log P_\theta(\epsilon_t|x_{t+1}). \quad (3)$$

An important tool with diffusion models is **Negative Prompting** [29, 1, 23, 65]. We can rewrite Equation 3 as

$$\log \widehat{P}_\theta(\epsilon_t|x_{t+1}, c) = \log P_\theta(\epsilon_t|x_{t+1}) + \gamma (\log P_\theta(\epsilon_t|x_{t+1}, c) - \log P_\theta(\epsilon_t|x_{t+1})) \quad (4)$$

Aside from its probabilistic interpretation, this equation also represents a vector arithmetic operation in latent space: we take a step of size  $\gamma$  away from the unconditional vector in the direction of the conditioning. Semantic vector linear arithmetic has proven to be effective in many situations in vision: striking examples have been generated by interpolations in GANs or diffusion models [47, 75, 14].

Moreover, the initial point does not have to be the unconditional latent, but any representation we want to move away from. We can introduce the "negative conditioning" or "negative prompt"  $\bar{c}$ , as well as a generalized equation resulting in Equation 3 when  $\bar{c} = \emptyset$ :

$$\log \widehat{P}_\theta(\epsilon_t|x_{t+1}, c, \bar{c}) = \log P_\theta(\epsilon_t|x_{t+1}, \bar{c}) + \gamma (\log P_\theta(\epsilon_t|x_{t+1}, c) - \log P_\theta(\epsilon_t|x_{t+1}, \bar{c})) \quad (5)$$

## 2.2 Classifier-Free Guidance of Language Models

To apply Classifier-Free Guidance to language models, we first have to define the semantic space to operate in. As demonstrated in [51, 60] and [27, 61], word embeddings and sentence embeddings have strong semantic structures. This makes the logits of token predictions a good choice of our latent space, due to its linear relationship with the last hidden layer. Using the logits avoids network editing [9] and is architecture agnostic.

Next, we need to define what is considered conditioning,  $c$ , in decoder-only language models. In the common situations, a user provides a *prompt*  $c$  which can be a context, an instruction, or the beginning of some text, and uses a language model to sample a sequence of continuation tokens  $w_i$  for the prompt  $c$ . Since a good continuation is expected to highly correlate to the prompt, we consider the prompt as our conditioning.

Similarly to Classifier Guidance [24, 84, 76], we wish to generate a text  $w$  which has a high likelihood of starting with  $c$ . We define the  $\gamma$ -reweighted distribution  $\widehat{P}(w|c) \propto P(w) \cdot P(c|w)^\gamma$ , and approximate it with CFG as  $\widehat{P}(w|c) \propto \frac{P(w|c)^\gamma}{P(w)^{\gamma-1}}$

In the case of autoregressive language models modeling  $P_\theta(w) = \prod_i^N P_\theta(w_i|w_{j < i})$ , we can unroll the formulation and obtain Equation 2 again:

$$\widehat{P}_\theta(w|c) \propto \prod_{i=1}^T \widehat{P}_\theta(w_i|w_{j < i}, c) \propto \prod_{i=1}^T \frac{P_\theta(w_i|w_{j < i}, c)^\gamma}{P_\theta(w_i|w_{j < i})^{\gamma-1}} \propto \frac{P_\theta(w|c)^\gamma}{P_\theta(w)^{\gamma-1}} \quad (6)$$

While conditioned diffusion models cannot predict unconditioned distributions without extra training, language models handle both  $P_\theta(w|c)$  and  $P_\theta(w)$  naturally due to being trained on finite context windows. Being able to drop the prefix  $c$  is a natural feature. We can thus sample the next  $i$ -th token  $w_i$  in the logits space:

$$\log \widehat{P}_\theta(w_i|w_{j < i}, c) = \log P_\theta(w_i|w_{j < i}) + \gamma(\log P_\theta(w_i|w_{i < j}, c) - \log P_\theta(w_i|w_{j < i})) \quad (7)$$

This formulation can be extended to accomodate “negative prompting”, as in Equation 5. Negative prompting as applied in autoregressive LMs will be further addressed in Section 3.4. Now, we will continue on to the next section, where we introduce our experiments. In this section, we will explore the effects of CFG on different variations of prompting.

### 3 Experiments

In this section we show that Classifier-Free Guidance reliably boosts performance across a variety of common prompting approaches. In Section 3.1 we show that CFG boosts zero-shot performance on a variety of standard NLP benchmarks, including achieving state-of-the-art performance on LAMBADA with LLaMa-7B. In Section 3.2 we apply CFG to *Chain-of-Thought prompts* [55, 82] an approach to allows the model to reason first before answering the question. Next, we test the performance of CFG on *text-to-text generation prompts* in Section 3.3. Finally, we show in Section 3.4 that CFG can be applied to *assistant* prompts (i.e. prompts with system-instructions).

#### 3.1 Basic Prompting: Zero-Shot Prompts

To test *basic, zero-shot prompting*, we consider a suite of zero-shot benchmarks implemented in the Language Model Evaluation Harness [33], which includes close-book QA [5, 39], common sense reasoning tasks [85, 69, 18, 12, 20, 8, 19], and sentence completion-tasks [58]. In these settings, the desired completions are short (often 1-2 tokens), so risks of meandering [76] or degradation [38] are low. We hypothesize that the main impact of CFG in these settings will be to reduce variance in output choices, as we explore more in Section 5.

We evaluate the GPT-2 model family[62], the Pythia model family [11] and the LLaMA model family[78] using different guidance strengths across a range of standard NLP benchmarks using EleutherAI’s Language Model Evaluation Harness [33] and implement CFG by starting the unconditional prompt at the last token of the initial prompt. The results are shown in Table 2. For better visualization, the charts for the GPT2 models, the Pythia models and the LLaMA models over the standard benchmarks are also shown in Figure 8, 9, and 10, respectively. We observe that except ARC (challenge) and Winogrande, the boost of performances from CFG is nontrivial and consistent. The reasons for these discrepancies are still unknown.

Furthermore, we note that even the smallest LLaMA 7B model achieves 81% accuracy in Lambada (OpenAI) zero-shot benchmark with  $\gamma = 1.5$ , outperforming the current SOTA (zero-shot) of PALM-540B (77.9%). Despite the fact that CFG almost doubles the computation during inference, the comparison is still noteworthy given that other models with comparable performances on Lambada (OpenAI) have much more parameters and would still require more compute than LLaMA 7B with CFG. Taken together, we show that CFG increases performance in basic prompting settings significantly.

#### 3.2 Deliberative Prompting: Chain-of-Thought

A variation on *basic prompting* has emerged recently called *Chain-of-Thought (CoT) prompting* [82]. In this setting, the model is prompted to generate a series of reasoning steps before giving an answer to the task: i.e.  $p(w_{cot}, w_a|w_p)$ , where  $w_{cot} = w_{p+1} \dots w_{c-1}$  and  $w_a$  is the answer.  $w_{cot}$  is designed to mimic the human reasoning or deliberation process. CoT has been shown to perform well in complex reasoning tasks that can not be fully addressed by model- or data-scaling [63], however, as observed by [82], long reasoning chains can diverge and either do not generate correct answers, or do not follow the expected result structure given by the prompt.

	ARC-c	ARC-e	BoolQ	HellaSwag
GPT2-small	22.7 / <b>23.0</b>	39.5 / <b>42.1</b>	48.7 / <b>57.0</b>	31.1 / <b>31.9</b>
GPT2-medium	<b>25.0</b> / 23.9	43.6 / <b>47.6</b>	58.6 / <b>60.1</b>	39.4 / <b>40.9</b>
GPT2-large	<b>25.1</b> / 24.7	46.6 / <b>51.0</b>	60.5 / <b>62.1</b>	45.3 / <b>47.1</b>
GPT2-xl	28.5 / <b>30.0</b>	51.1 / <b>56.5</b>	61.8 / <b>62.6</b>	50.9 / <b>52.4</b>
Pythia-160M	<b>23.5</b> / 23.0	39.5 / <b>42.2</b>	55.0 / <b>58.3</b>	30.1 / <b>31.2</b>
Pythia-410M	<b>24.1</b> / 23.8	45.7 / <b>50.3</b>	60.6 / <b>61.2</b>	40.6 / <b>41.6</b>
Pythia-1B	27.0 / <b>28.0</b>	49.0 / <b>54.9</b>	60.7 / <b>61.8</b>	47.1 / <b>48.9</b>
Pythia-1.4B	28.6 / <b>29.6</b>	53.8 / <b>59.6</b>	63.0 / <b>63.8</b>	52.1 / <b>54.3</b>
Pythia-2.8B	33.1 / <b>34.5</b>	58.8 / <b>65.4</b>	64.7 / <b>64.7</b>	59.3 / <b>61.9</b>
Pythia-6.9B	35.2 / <b>36.1</b>	61.3 / <b>67.4</b>	63.7 / <b>64.6</b>	64.0 / <b>66.5</b>
Pythia-12B	36.9 / <b>38.7</b>	64.1 / <b>72.6</b>	67.6 / <b>67.8</b>	67.3 / <b>69.6</b>
LLaMA-7B	41.5 / <b>43.9</b>	52.5 / <b>58.9</b>	<b>73.1</b> / 71.8	73.0 / <b>76.9</b>
LLaMA-13B	47.8 / <b>54.2</b>	74.8 / <b>79.1</b>	<b>78.0</b> / 75.8	79.1 / <b>82.1</b>
LLaMA-30B	52.9 / <b>57.4</b>	78.9 / <b>83.2</b>	<b>82.7</b> / 80.0	82.6 / <b>85.3</b>
LLaMA-65B	55.6 / <b>59.0</b>	79.7 / <b>84.2</b>	<b>84.8</b> / 83.0	84.1 / <b>86.3</b>

(a)

	PIQA	SCIQ	TriviaQA	WinoGrande	Lambada
GPT2-small	62.5 / <b>63.8</b>	64.4 / <b>70.8</b>	5.5 / <b>6.5</b>	<b>51.6</b> / 50.5	32.6 / <b>44.6</b>
GPT2-medium	66.4 / <b>66.9</b>	67.2 / <b>76.7</b>	8.3 / <b>9.3</b>	<b>53.1</b> / 52.1	43.0 / <b>55.8</b>
GPT2-large	69.2 / <b>70.2</b>	69.4 / <b>78.8</b>	11.1 / <b>12.0</b>	<b>55.4</b> / 54.4	47.7 / <b>60.5</b>
GPT2-xl	70.5 / <b>71.3</b>	76.1 / <b>82.4</b>	14.7 / <b>15.2</b>	<b>58.3</b> / 55.6	51.2 / <b>62.5</b>
Pythia-160M	61.4 / <b>62.1</b>	67.0 / <b>75.4</b>	4.1 / <b>5.3</b>	<b>52.3</b> / 51.1	32.8 / <b>47.4</b>
Pythia-410M	67.1 / <b>67.8</b>	72.1 / <b>79.0</b>	7.9 / <b>9.1</b>	<b>52.9</b> / 50.7	51.3 / <b>64.0</b>
Pythia-1B	69.2 / <b>70.5</b>	76.0 / <b>82.9</b>	<b>12.3</b> / 12.3	<b>53.9</b> / 51.5	56.2 / <b>69.0</b>
Pythia-1.4B	71.1 / <b>72.5</b>	79.4 / <b>85.1</b>	15.9 / <b>15.9</b>	<b>57.4</b> / 56.0	61.6 / <b>72.7</b>
Pythia-2.8B	73.6 / <b>75.8</b>	83.3 / <b>88.2</b>	<b>22.1</b> / 20.9	<b>60.1</b> / 57.9	64.6 / <b>76.5</b>
Pythia-6.9B	76.3 / <b>77.4</b>	84.3 / <b>89.7</b>	<b>28.2</b> / 27.2	<b>61.1</b> / 60.3	67.1 / <b>78.8</b>
Pythia-12B	77.0 / <b>78.4</b>	87.7 / <b>91.9</b>	33.4 / 32.1	<b>65.0</b> / 63.4	70.4 / <b>80.6</b>
LLaMA-7B	77.4 / <b>79.8</b>	66.3 / <b>75.4</b>	<b>56.0</b> / 52.7	<b>67.1</b> / 65.5	73.6 / <b>81.3</b>
LLaMA-13B	80.1 / <b>80.9</b>	91.1 / <b>95.1</b>	<b>62.4</b> / 59.8	<b>72.8</b> / 71.5	76.2 / <b>82.2</b>
LLaMA-30B	<b>82.3</b> / 82.3	94.3 / <b>96.4</b>	<b>69.7</b> / 67.9	<b>75.8</b> / 74.1	77.5 / <b>83.9</b>
LLaMA-65B	82.3 / <b>82.6</b>	95.1 / <b>96.6</b>	<b>73.3</b> / 71.8	<b>77.4</b> / 76.1	79.1 / <b>84.0</b>

(b)

Figure 2: Results of general natural language benchmarks. In each cell, the first value is the result for  $\gamma = 1$  (baseline) and the second value is the result for  $\gamma = 1.5$  (ours). LLaMA 7B with CFG on Lambada zero-shot already outperforms vanilla PaLM 540B, Chinchilla 70B, and GPT-3 175B, tops the SOTA leaderboard for Lambada zero-shot as of June 26th, 2023

This setting poses a variation on the prior *base-case* setting: now, the continuation  $w_c = [w_{cot}, w_a]$  is expected to be longer than 1-2 tokens. We hypothesize that compared to basic zero-shot prompting explored in Section 3.1, CFG will also be able to enforce better reasoning chains with less drift.

We evaluate the effectiveness of our proposed CFG method with respect to chain-of-thought prompting on two arithmetic reasoning tasks: GSM8K [21] and AQuA [48]. We follow [80] few-shot prompt and parsing setting, with respect to two open source LLM models: WizardLM-30B [83] and Guanaco-65B [25]. As can be seen in Figure 3, 15, using CFG increases the percentage of CoT which results in a valid answer that could be parsed. For low guidance strengths, this results in boosting the model performances. However, for large values, although the model returns more valid results, the quality of the chains is also impacted, and overall the model performances degrade. A qualitative comparison is provided in Table 15, 14.

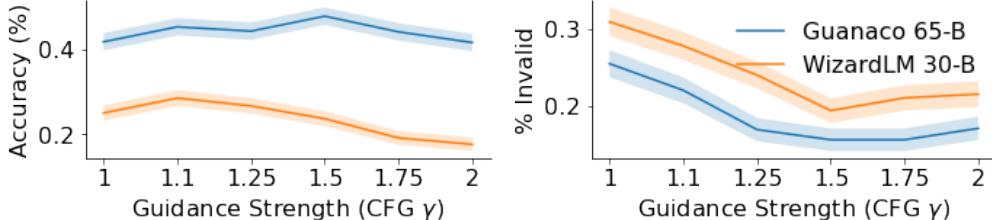


Figure 3: CFG impact on chain-of-thought prompting with respect to GSM8K dataset. For small CFG values, using CFG increases the percentage of chains which end in a valid answer structure while increasing the model accuracy. For large values the invalid percentage remains small but the accuracy drop.

We have only scratched the surface of exploring CFG’s interactions with CoT; for instance, instead of upweighting just  $w_p$ , we might upweight  $w_p, w_{cot}$ , or other variations. We anticipate in future work being able to more fully test variations of CFG-weighting on different parts of the CoT process.

### 3.3 Text-to-Text Prompts: Generation

In contrast to *basic prompting* and *CoT-prompting*, where we ultimately expect a short answer,  $w_a$ , many settings require lengthier continuations. In this section, we study a prompt setting where the quality of answers are highly dependent on the ability to stay on target over long sequences of text (both prompt,  $w_p$  and continuation,  $w_c$ ). Here we focus on code generation, and in Appendix D.1 we report results on machine translation. We hypothesize that, in contrast to Sections 3.1 and 3.2, these tasks require longer-form completions, which Classifier-Free Guidance’s effectiveness in enforcing adherences to many different parts of the prompt.

#### 3.3.1 Program synthesis evaluations

Computer programs represent an important language-modeling case, as formal language differs from natural language in many ways including the use of well-defined structures. Testing Classifier-Free Guidance on code-related tasks improves the robustness of our hypothesis over different distributions of data. In the exploratory experiments, we prompt GPT-J [79] and CodeGen-350M-mono [54] for small-scale code generations and observe positive results (see Appendix D.2). And then we perform a thorough evaluation on the HumanEval benchmark [16].

#### 3.3.2 HumanEval benchmark

To systematically investigate the impact of Classifier-Free Guidance on code completion abilities, we evaluate models using different CFG strengths on HumanEval benchmark [16]. HumanEval benchmark contains 164 coding tasks in Python where the prompts are given by a function signature and a docstring. The model will generate continuations of the prompt, and the resulting programs will be tested against a set of unit tests for each task which evaluate the correctness of Python programs. We choose CodeGen-350M-mono, CodeGen-2B-mono and CodeGen-6B-mono ([54]) which are specialized in Python program synthesis.<sup>1</sup>

Various CFG strengths<sup>2</sup> are tested on 3 different temperatures 0.2, 0.6, 0.8 with the evaluation metrics being pass@ $k$  for  $k = 1, 10, 100$ <sup>3</sup>. Here we show the results for temperature= 0.2 in Table 2. The full results are summarized in Appendix C.3 in Table 5, 6 and 7 and Figure 12, 13 and 14.

We observe that low CFG ( $\gamma \leq 1.5$ ) increases the pass@1 rate uniformly<sup>4</sup>. High CFG ( $\gamma \geq 1.5$ ) leads to a deterioration of performance. We also note that the improvement from CFG diminishes or harms performance at pass@ $k$  at high  $k$ .

To further investigate the effect of CFG, we break down the pass@1 evaluations on CodeGen-350M-mono for  $\gamma = 1, 1.25$  task-by-task<sup>5</sup>. We notice that the number of tasks where CFG outperforms is still more than the one where CFG underperforms for all temperatures 0.2, 0.6, 0.8 (See Table 4).

<sup>1</sup>Note: CodeGen-16B-mono is omitted due to the compute constraint.

<sup>2</sup> $\gamma = 1.0, 1.1, 1.25, 1.5, 1.75, 2.0$

<sup>3</sup>The definition of pass@ $k$  according to [16]: “ $k$  code samples are generated per problem, a problem is considered solved if any sample passes the unit tests, and the total fraction of problems solved is reported.”

<sup>4</sup>Note that the effect of low CFG on the pass@1 rate is consistent with the results of the general benchmarks in the previous section.

<sup>5</sup>See the scatter plot at temperature 0.2, 0.6, 0.8 in appendix, Figure 15a, 15b, 15c

$\gamma$	CodeGen-350M			CodeGen-2B			CodeGen-6B		
	k=1	k=10	k=100	k=1	k=10	k=100	k=1	k=10	k=100
1.0	11.0%	17.0%	<b>22.0%</b>	19.5%	25.5%	<b>29.8%</b>	19.5%	25.5%	<b>29.8%</b>
1.1	<b>11.8%</b>	<b>18.1%</b>	20.1%	20.4%	25.4%	28.0	20.4%	25.4%	28.0%
1.25	11.4%	17.3%	18.9%	19.7%	25.4%	28.0	19.7%	25.4%	28.0%
1.5	10.9%	16.7%	18.3%	<b>20.9%</b>	<b>26.7%</b>	29.2%	<b>20.9%</b>	<b>26.7%</b>	29.2
1.75	10.3%	16.0%	18.2%	20.4%	26.2%	28.6%	20.4%	26.2%	28.6%
2.0	8.6%	14.6%	17.6%	16.5%	22.4%	24.4%	16.5%	22.4%	24.4%

Table 2: CodeGen results with temperature = 0.2. CFG in nearly all cases increases performance, but the optimal  $\gamma$  value varies.

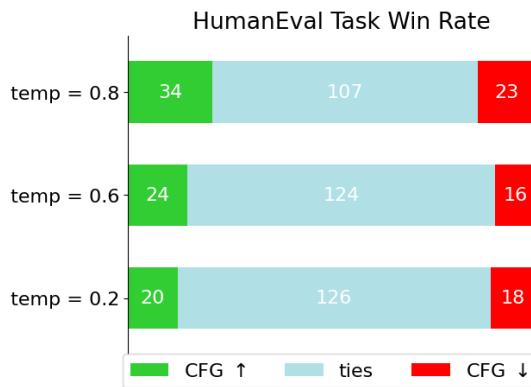


Figure 4: HumanEval task count comparison between  $\gamma = 1, 1.25$  for CodeGen-350M-mono

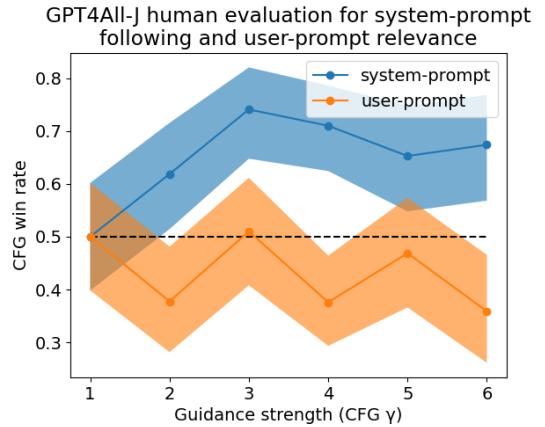


Figure 5: Evaluators (611 votes, 71 unique voters) significantly preferred the system-prompt with CFG (max at  $\gamma = 3$ ). The user-prompt relevance, not subject to CFG, did not degrade until  $\gamma \geq 4$ , showing a clear win without tradeoff at  $\gamma = 3$ .

We also find that without CFG, many tasks exhibit small nonzero passing rates while having 0% rate with CFG. This explains the decreasing improvement of CFG in pass@ $k$  for large  $k$ , as larger  $k$  significantly boosts the passing rate of difficult tasks where the rates are low but nonzero.

Overall, the consistent improvement on pass@1 rates and the reduced effect on pass@100 rates support our hypothesis that CFG strengthens the adherence to the prompt at the small cost of reduced variability and creativity.

### 3.4 Negative Prompting: Improving Assistants

Finally, we explore an addition to Classifier-Free Guidance called *negative prompting*. With negative prompting, the user specifies what they do *not* want in the output (e.g. “low resolution”, “bad hands, bad anatomy, amateur drawing” in text-to-image), which is then used to improve generation quality.

We explore this idea in the context of chatbots. Chatbots give us a setting where the *prompt* is expanded into a *multi-stage prompt*<sup>6</sup>. In chatbots, the language model is prompted with a two-part prompt: (1) the instruction,  $w_s$  (sometimes called “system prompt”) which may give contextual information (e.g. the “current date”), or behavioral guidelines (e.g. style, alignment, persona, etc.); and (2)  $w_p$ , the user-prompt, or the user’s query. See Table 1 for an example. Adherence becomes an even greater challenge, as our initial explorations shows. We observe systems like Alpaca [77, 59, 3] often ignore changes to their default system-prompt, and may even expose models to attacks like prompt injection [36].

<sup>6</sup>We note that this extension to *basic-prompting* stands as a mirror to *CoT-prompting*’s extension (Section 3.2). In *CoT-prompting*, the *continuation* is expanded to a *multi-stage completion*; here, the *prompt* is expanded.

We explore CFG with negative prompting to increase the success rate of different system prompts. We set the negative prompt  $\bar{c}$  to be the default system-prompt for the models we use (i.e. “The prompt below is a question to answer, a task to complete, or a conversation to respond to; decide which and write an appropriate response.”) and set  $c$  to be the edited prompt (e.g. “The prompt below is a question to answer, a task to complete, or a conversation to respond to; decide which and write *a sad* response.”). This approach not only makes the sampling more prompt-aware in general, but directly emphasizes the difference between *our* system-prompt and the model’s default system-prompt.

To test this approach with chatbots, we generate system-prompts,  $n_c = 25$ , and user-prompts,  $n_p = 46$ , and sample 1740 random combinations of them. An example is shown in Table 1 (in Appendix G we include the full list of  $c$  and  $p$  we use). We use GPT4All-J v1.3-jazzy to generate two completions for each sampled combination: the first is sampled without CFG, and the second is sampled with CFG, with a guidance strength randomly chosen  $\in \{1, 2, 3, 4, 5, 6\}$ . Our hypothesis is that CFG increases system-prompt following, ideally without hurting the relevance to the user input.

We run a human preference study on our sampled continuations, where participants are shown both, blindly, and asked to assess two things: A. which output better follows the system-prompt,  $c$  and B. which output better follows the user-prompt  $p$ . Our results in Figure 5 shows compelling evidence that CFG emphasized the difference between  $c$  and  $\bar{c}$  more than sampling with  $c$  alone. There is a clear peak at  $\gamma = 3$  with 75% of system-prompt following preference over  $\gamma = 1$  and undegraded user-prompt relevance (52%).

## 4 Computational Cost Analysis

In the previous section we showed improvements across a wide array of benchmarks and contexts. However, since classifier-free guidance requires two passes through the network, users who are compute-constrained rather than VRAM constrained might wonder if CFG is interesting to them at all, and if they should not run a model twice as big instead.

To answer this question, we calculate the FLOP for each of the benchmark experiments that we ran in Section 3.1. We then compare across model sizes, with and without CFG. We conclude with the surprising finding that, across 5 out of 9 tasks, there is a statistically *insignificant difference* between using CFG and using vanilla prompting with a model of twice the size at  $p = .01$ , according to ANCOVA regression analysis [67]. Of the significantly different tasks, 2 favor CFG and 2 favor vanilla. See Appendix C.2, specifically Figure 11, for more details.

In other words, and most significantly, this indicates that, overall, a model using CFG can generally perform just as well as a model twice as large. This has enormous implications for training budgets and inference latency due to limited VRAM usage, which we seek to explore in future work.

## 5 Explaining the Success of Classifier-Free Guidance

In this section, we try to derive insights on the impact of Classifier-Free Guidance on generation, both quantitatively and qualitatively. We sample a dataset of 32,902 datapoints from the P3 dataset [70] and use the Falcon-7b-Base model family [2] as an exploratory model. Our goal is to analyze the logit distributions – we describe how in the following sections. Many of our comparisons are done with reference to an instruction-tuned model, for which we use the Falcon-7b-Instruct version. We replicate our findings on other models and datasets as well: the Open-Assistant Dataset [42] and Redpajama-3b model family<sup>7</sup>.

### 5.1 Classifier-Free Guidance’s Effect on Sampling Entropy

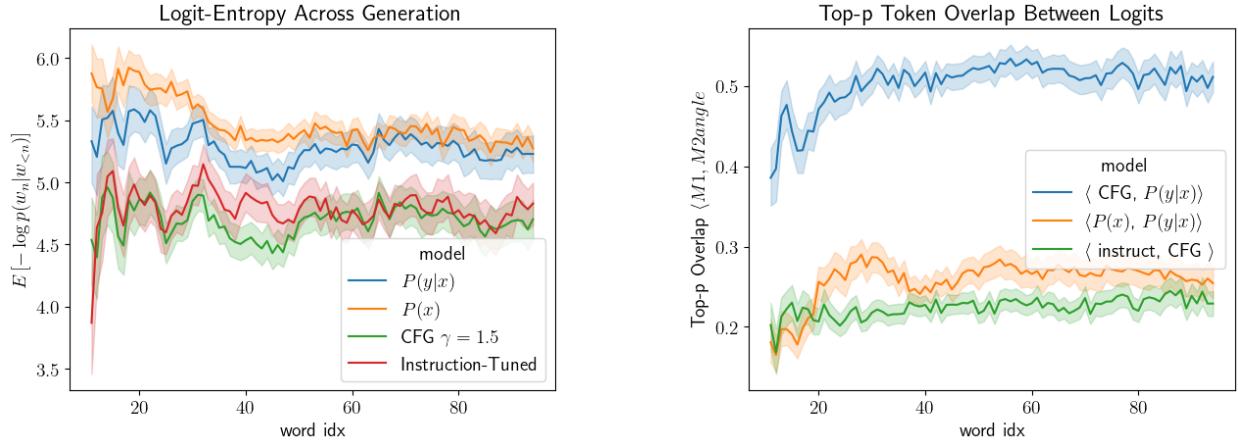
We suspect that CFG, by focusing  $P(y|x)$  on the prompt, will reduce the entropy of the logit distribution. CFG entropy distribution is significantly lower across generation time-steps vanilla prompting, with a mean of 4.7 vs. 5.4. (See Figure 6a). The effect this has is to restrict the number of tokens in the top-p=90% of the vocabulary distribution (See in Figure 6b). We do observe qualitatively, shown in Section 5.3, that the top tokens do not shift too much, but they do re-order to some extent, which shows that CFG is not simply having the same effect as the temperature parameter.

### 5.2 CFG’s Relation to Instruction Tuning

Our next question: *how* is Classifier-Free Guidance affecting the vocabulary distribution? We attempt to answer this question quantitatively, hypothesizing that CFG has similar effects to instruction-tuning, which we assume trains a model to focus on the prompt. We find that both CFG and Instruction-Tuned model variants have similar entropies

---

<sup>7</sup><https://www.together.xyz/blog/redpajama>



(a) Entropy of logits for the vanilla prompted distribution  $P(y|x)$ , the unprompted distribution,  $P(x)$ , the CFG- $\gamma = 1.5$  distribution and an instruction-tuned model  $P_{\text{instruct}}(y|x)$ .

(b) Number of tokens overlapping in top-p=90% of vocabulary distributions between that of: CFG, that of the vanilla prompted model,  $p(y|x)$ , and that of the unprompted model,  $P(x)$ .

Figure 6: We show into how CFG alters the logit distribution of the vanilla prompted model,  $P(y|x)$ . CFG lowers the entropy to a level roughly similar to instruction-tuned model variant. CFG shares roughly 50% of the tokens in top-p=0.9 as the vanilla  $P(y|x)$  model.

	PPL $p(y x)$	PPL cfg	PPL instruct
PPL $p(y x)$	1.0	0.94	0.83
PPL cfg	0.94	1.0	0.7
PPL instruct	0.83	0.7	1.0

(a) Correlation between the perplexities of each model on P3.

	$r_s$ (sim)	p-val.
PPL $p(y x)$	0.01	0.2
PPL cfg	-0.04	<.001
PPL instruct	0.04	<.001

(b) Correlation between the perplexity and similarity between Instruction-Tuned and CFG.

Figure 7: We seek to identify *when* CFG is similar to instruction-tuning. Models mostly agree on the difficulty of input sentences, and in cases where they do not, CFG and Instruction-tuned have similar top-p overlaps.

across generation samples. However, as shown in Figure 6b the vocabulary distributions across our samples are largely not overlapping.

We find that, overall, our hypothesis about the similarity is wrong: CFG is not having a similar effect on the vocabulary logits as instruction-tuning. To explore, we seek to derive insight from edge-cases where it does. We look for characteristics to explain when CFG is similar to Instruction-Tuning (in terms of top-p overlap). One case pops out: when the prompt is longer, CFG agrees more – we observe a significant spearman correlation of  $r_s = .05$  between prompt-length and Instruction/CFG agreement. We also observe small but significant correlations between perplexity and agreement. As shown in Table 7, harder phrases for Instruction-Tuned models are typically where CFG and Instruction-Tuned models align. We conclude that CFG is altering the model in ways that might complement instruction-tuning, opening the door to future explorations.

### 5.3 Visualizing Classifier-Free Guidance

Finally, we provide qualitative insights into the reordering of the vocabulary, after Classifier-Free Guidance is applied. We note that the Equation can be rewritten as

$$\log P_\gamma(w_t|w_{<t}, c) = \log P(w_t|w_{<t}, \bar{c}) + \gamma(\log P(w_t|w_{<t}, c) - \log P(w_T|w_{<t}, \bar{c})) \quad (8)$$

We propose, at each timestep, to visualize the vocabulary ranked by the difference  $\log P(w_t|w_{<t}) - \log P(w_T|w_{<t})$ . This shows the impact of the method, qualitatively, by revealing the tokens that are encouraged or discouraged the

current	top1	top2	top3	top4	top5	...	bottom5	bottom4	bottom3	bottom2	bottom1
France	flipping	destroying	waking	stopping	causing	...	guiName	ufact	Outs	kees	"].,"
, landing	crashing	landing	swoop	plummet	soDeliveryDate	POLIT	Occupations	568	publishes	hostages	
on Buildings	neigh	invis	atop	overhead	omin	quotas	Russo	Germans	passports	DragonMagazine	
Notre Dame Cathedral	Basil	Mos	Cathedral	Cheong	Plaza	...	MFT	č	urse	arb	
.	Catederal	monument	cathedral	Basil	Mosque	Eugene	alach	aila	voy	aund	wk
It swoop	Dragon	dragons	dragon	Dragon	Dragons	1915	1914	1944	1934	1913	Wanted
circled skysc	circled	pedestrians	architectural	dart	hopped	bolted	concludes	reads	reads	culmin	marks
Paris night	amura	rum	slowing	hanging	skyline	Newfoundland	Ukrain	Zamb	Johnston	Queensland	
a longer	while	long	blocking	anim	animate	prematurely	capit	bombed	Mé	owing	
bit longer	MORE	awhile	ortex	animate	length	ims	chin	chel	ille	ller	
, started	feathers	dragon	dragon	more	prematurely	hof	nw	ari	trop		
and dragon	dragons	golden	Winged	dragons	dragons	inväl	Junction	Palest	endas	CVE	
then dragon	DRAG	dragons	neigh	DRAGON	dragons	CVE	ival	Ukrain	onet	Commodore	
flew ukong	skelet	rum	swoop	acles	RG	onet	Kear	TPS	Tags		
over rium	Rockefeller	Plaza	Times	Symphony	Brittany	Newfoundland	NJ	444	programmes		
the Griffith	Zeus	Hag	Science	Raphael	Midlands	Balt	isconsin	Yugoslavia			
E BI	Rowe	ident	Methodist	allah	coasts	frontier	deserts	Balkans			
iff Armory	Library	restrooms	Mansion	Mahmoud	indo	ento	seys	Desire			
el restaurant	Middle	restroom	boutique	museum	iband	Off	itime	Norm			
Tower Property	omin	Foundation	Creature	>"	gee	thence	centres	detach	rift		
. Then dragons	dragon	Dragons	Dragon	DRAGON	1944	1942	Instrument	Balt	1943		
it dragons	dragon	dragon	Dragons	Dragon	Manz	Hopkins	Instrument	Squadron			
flew over upro	ukong	neigh	Winged	Draco	CVE	udder	services	obbies			
	Chinatown	Financial	Spider	tallest	INC	inary	lein	quickShip	CVE		
					warr						

Table 3: Given the prompt **The dragon flew over Paris, France** we display, at each sampling step, the vocabulary ranked for  $P(w_t|w_{<t}) - \log P(w_T|\hat{w})$  for the next step. We can see CFG encouraging tokens about **flying dragons** and **Paris**, and discouraging other **topics or regions**

most. In Figure 3, we prompt a model with  $c = \text{"The dragon flew over Paris, France"}\bar{c} = \emptyset$  and observe that tokens about dragons and Paris get upweighted while tokens about other locations (“Queensland”), dates (“1913”), or topics (“hostages”, “voyages”) get downweighted. This confirms our initial assumptions, as we observe CFG encouraging tokens related to and discourages tokens unrelated to the prompt.

We find this visualization approach to be a useful prompt engineering tool, by using the new prompt under testing as  $c$  and setting  $\bar{c}$  as the current baseline prompt. The visualization shows the differential impact over the whole vocabulary on the next token prediction, in an interpretable way.

## 6 Conclusion

We have shown that Classifier-Free Guidance, which was originally conceived of in text-to-image applications, can be an effective way of increasing adherence to the prompt in autoregressive language modeling. In contrast to text-to-vision, CFG in autoregressive language modeling works out-of-the-box, without the need to further train the model. We have shown that CFG can boost performance across an array of canonical benchmarks in NLP that involve variations of the prompt: *basic prompting*, *chain-of-thought prompting*, *text-to-text prompting* and *chatbot prompting*. Finally, we sought to explain the effects of CFG by showing it decreased sampling entropy, but not in the same ways that Instruction-tuned models do. Ultimately, we leave for future work the exact effects that CFG is having, but we propose qualitative visualizations that confirm our intuitions around prompt adherence.

Our work also integrates into a growing body of inference techniques aimed at perturbing the logit distributions of an LM [45, 73]. We demonstrate that by doubling the inference FLOP using CFG brings performances of a model about twice the size. This allows training smaller models, which can be ran on smaller hardware, and are cheaper to train.

Our work faces the following limitations: CFG requires tweaking and exploration:  $\gamma$  values that might work in one context (i.e. long-form generation) might be poorly suited for another context. It’s also possible that CFG might be misused. We have not tested the effects of CFG if used in conjunction with malicious strategies for hacking language models, including but not limited to: prompt injection and prompts aimed at overriding alignment. It’s possible that there are unforeseen effects induced by an increased adherence to parts of the prompt. We tried to explore this at length, both quantitatively and qualitatively, and we designed tasks that might reveal such behavior. However, we cannot conclude this method is risk-free. We advocate for standardized benchmarks aimed more squarely at language-model risk (including, possibly, pairs of models along with known prompt injections). Such standardized benchmarks could help us unit-test an advancement like CFG before releasing it into the wild.

## Acknowledgements

We are grateful to Stability and CoreWeave for providing the compute to run the evaluations.

We also thank the volunteers who took part in the GPT4All experiment.

Alexander Spangher would like to thank Bloomberg News for a 4 year PhD fellowship that generously funds his research.

## References

- [1] How does negative prompt work? <https://stable-diffusion-art.com/how-negative-prompt-work/>.
- [2] E. Almazrouei, H. Alobeidli, A. Alshamsi, A. Cappelli, R. Cojocaru, M. Debbah, E. Goffinet, D. Heslow, J. Launay, Q. Malartic, B. Noune, B. Pannier, and G. Penedo. Falcon-40B: an open large language model with state-of-the-art performance. 2023.
- [3] Y. Anand, Z. Nussbaum, B. Duderstadt, B. Schmidt, and A. Mulyar. Gpt4all: Training an assistant-style chatbot with large scale data distillation from gpt-3.5-turbo. <https://github.com/nomic-ai/gpt4all>, 2023.
- [4] A. Askell, Y. Bai, A. Chen, D. Drain, D. Ganguli, T. Henighan, A. Jones, N. Joseph, B. Mann, N. DasSarma, et al. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021.
- [5] S. Auer, D. A. Barone, C. Bartz, E. G. Cortes, M. Y. Jaradeh, O. Karras, M. Koubarakis, D. Mouromtsev, D. Pliukhin, D. Radyush, et al. The sciqa scientific question answering benchmark for scholarly knowledge. *Scientific Reports*, 13(1):7240, 2023.
- [6] Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.
- [7] R. Barzilay and M. Lapata. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34, 2008.
- [8] K. Basu, F. Shakerin, and G. Gupta. Aqua: Asp-based visual question answering. In *Practical Aspects of Declarative Languages: 22nd International Symposium, PADL 2020, New Orleans, LA, USA, January 20–21, 2020, Proceedings 22*, pages 57–72. Springer, 2020.
- [9] N. Belrose, D. Schneider-Joseph, S. Ravfogel, R. Cotterell, E. Raff, and S. Biderman. Leace: Perfect linear concept erasure in closed form. *arXiv preprint arXiv:2306.03819*, 2023.
- [10] S. Biderman and E. Raff. Fooling moss detection with pretrained language models. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 2933–2943, 2022.
- [11] S. Biderman, H. Schoelkopf, Q. Anthony, H. Bradley, K. O’Brien, E. Hallahan, M. A. Khan, S. Purohit, U. S. Prashanth, E. Raff, A. Skowron, L. Sutawika, and O. van der Wal. Pythia: A suite for analyzing large language models across training and scaling, 2023.
- [12] Y. Bisk, R. Zellers, J. Gao, Y. Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439, 2020.
- [13] O. Bojar, C. Buck, C. Federmann, B. Haddow, P. Koehn, J. Leveling, C. Monz, P. Pecina, M. Post, H. Saint-Amand, R. Soricut, L. Specia, and A. s. Tamchyna. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics.
- [14] A. Brock, T. Lim, J. Ritchie, and N. Weston. Neural photo editing with introspective adversarial networks. In *International Conference on Learning Representations*.
- [15] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [16] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba. Evaluating large language models trained on code. 2021.

- [17] J. Chorowski and N. Jaitly. Towards better decoding and language model integration in sequence to sequence models. *arXiv preprint arXiv:1612.02695*, 2016.
- [18] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.
- [19] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*, 2018.
- [20] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.
- [21] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.
- [22] K. Crowson, S. Biderman, D. Kornis, D. Stander, E. Hallahan, L. Castricato, and E. Raff. Vqgan-clip: Open domain image generation and editing with natural language guidance. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVII*, pages 88–105. Springer, 2022.
- [23] K. Crowson, S. Biderman, D. Kornis, D. Stander, E. Hallahan, L. Castricato, and E. Raff. Vqgan-clip: Open domain image generation and editing with natural language guidance. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVII*, pages 88–105. Springer, 2022.
- [24] S. Dathathri, A. Madotto, J. Lan, J. Hung, E. Frank, P. Molino, J. Yosinski, and R. Liu. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*, 2019.
- [25] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer. Qlora: Efficient finetuning of quantized llms, 2023.
- [26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805, 2019.
- [28] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.
- [29] Y. Du, S. Li, and I. Mordatch. Compositional visual generation with energy based models. *Advances in Neural Information Processing Systems*, 33:6637–6647, 2020.
- [30] V. K. Felkner, H.-C. H. Chang, E. Jang, and J. May. Towards winoqueer: Developing a benchmark for anti-queer bias in large language models. *arXiv preprint arXiv:2206.11484*, 2022.
- [31] Z. Fu, W. Lam, A. M.-C. So, and B. Shi. A theoretical analysis of the repetition problem in text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12848–12856, 2021.
- [32] R. Gal, O. Patashnik, H. Maron, G. Chechik, and D. Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. *arXiv preprint arXiv:2108.00946*, 2021.
- [33] L. Gao, J. Tow, S. Biderman, S. Black, A. DiPofi, C. Foster, L. Golding, J. Hsu, K. McDonell, N. Muennighoff, J. Phang, L. Reynolds, E. Tang, A. Thite, B. Wang, K. Wang, and A. Zou. A framework for few-shot language model evaluation, Sept. 2021.
- [34] S. Gehman, S. Gururangan, M. Sap, Y. Choi, and N. A. Smith. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. *arXiv preprint arXiv:2009.11462*, 2020.
- [35] F. A. Gers, J. Schmidhuber, and F. Cummins. Learning to forget: Continual prediction with lstm. *Neural computation*, 12(10):2451–2471, 2000.
- [36] K. Greshake, S. Abdehnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz. More than you've asked for: A comprehensive analysis of novel prompt injection threats to application-integrated large language models. *arXiv preprint arXiv:2302.12173*, 2023.
- [37] J. Ho and T. Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.
- [38] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

- [39] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*, 2017.
- [40] N. S. Keskar, B. McCann, L. R. Varshney, C. Xiong, and R. Socher. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.
- [41] G. Kim, T. Kwon, and J. C. Ye. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2426–2435, 2022.
- [42] A. Köpf, Y. Kilcher, D. von Rütte, S. Anagnostidis, Z.-R. Tam, K. Stevens, A. Barhoum, N. M. Duc, O. Stanley, R. Nagyfi, et al. Openassistant conversations—democratizing large language model alignment. *arXiv preprint arXiv:2304.07327*, 2023.
- [43] B. Krause, A. D. Gotmare, B. McCann, N. S. Keskar, S. Joty, R. Socher, and N. F. Rajani. Gedi: Generative discriminator guided sequence generation. *arXiv preprint arXiv:2009.06367*, 2020.
- [44] X. Li, J. Thickstun, I. Gulrajani, P. S. Liang, and T. B. Hashimoto. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343, 2022.
- [45] X. L. Li, A. Holtzman, D. Fried, P. Liang, J. Eisner, T. Hashimoto, L. Zettlemoyer, and M. Lewis. Contrastive decoding: Open-ended text generation as optimization. *arXiv preprint arXiv:2210.15097*, 2022.
- [46] S. Lin, B. Liu, J. Li, and X. Yang. Common diffusion noise schedules and sample steps are flawed, 2023.
- [47] H. Ling, K. Kreis, D. Li, S. W. Kim, A. Torralba, and S. Fidler. Editgan: High-precision semantic image editing. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.
- [48] W. Ling, D. Yogatama, C. Dyer, and P. Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada, July 2017. Association for Computational Linguistics.
- [49] P. Manakul, A. Liusie, and M. J. Gales. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*, 2023.
- [50] T. Meng, S. Lu, N. Peng, and K.-W. Chang. Controllable text generation with neurally-decomposed oracle. *arXiv preprint arXiv:2205.14219*, 2022.
- [51] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean. Efficient estimation of word representations in vector space. In *International Conference on Learning Representations*, 2013.
- [52] N. Muennighoff, T. Wang, L. Sutawika, A. Roberts, S. R. Biderman, T. L. Scao, M. S. Bari, S. Shen, Z. X. Yong, H. Schoelkopf, X. Tang, D. R. Radev, A. F. Aji, K. Almubarak, S. Albanie, Z. Alyafeai, A. Webson, E. Raff, and C. Raffel. Crosslingual generalization through multitask finetuning. *ArXiv*, abs/2211.01786, 2022.
- [53] A. Q. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. Mcgrew, I. Sutskever, and M. Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. In *International Conference on Machine Learning*, pages 16784–16804. PMLR, 2022.
- [54] E. Nijkamp, B. Pang, H. Hayashi, L. Tu, H. Wang, Y. Zhou, S. Savarese, and C. Xiong. Codegen: An open large language model for code with multi-turn program synthesis. In *The Eleventh International Conference on Learning Representations*, 2023.
- [55] M. Nye, A. J. Andreassen, G. Gur-Ari, H. Michalewski, J. Austin, D. Bieber, D. Dohan, A. Lewkowycz, M. Bosma, D. Luan, C. Sutton, and A. Odena. Show your work: Scratchpads for intermediate computation with language models. In *Deep Learning for Code Workshop*, 2022.
- [56] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.
- [57] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.
- [58] D. Paperno, G. Kruszewski, A. Lazaridou, Q. N. Pham, R. Bernardi, S. Pezzelle, M. Baroni, G. Boleda, and R. Fernández. The lambada dataset: Word prediction requiring a broad discourse context. *arXiv preprint arXiv:1606.06031*, 2016.
- [59] B. Peng, C. Li, P. He, M. Galley, and J. Gao. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*, 2023.

- [60] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing*, 2014.
- [61] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. Improving language understanding by generative pre-training. 2018.
- [62] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [63] J. W. Rae, S. Borgeaud, T. Cai, K. Millican, J. Hoffmann, F. Song, J. Aslanides, S. Henderson, R. Ring, S. Young, E. Rutherford, T. Hennigan, J. Menick, A. Cassirer, R. Powell, G. v. d. Driessche, L. A. Hendricks, M. Rauh, P.-S. Huang, A. Glaese, J. Welbl, S. Dathathri, S. Huang, J. Uesato, J. Mellor, I. Higgins, A. Creswell, N. McAleese, A. Wu, E. Elsen, S. Jayakumar, E. Buchatskaya, D. Budden, E. Sutherland, K. Simonyan, M. Paganini, L. Sifre, L. Martens, X. L. Li, A. Kuncoro, A. Nematzadeh, E. Gribovskaya, D. Donato, A. Lazaridou, A. Mensch, J.-B. Lespiau, M. Tsimpoukelli, N. Grigorev, D. Fritz, T. Sotiaux, M. Pajarskas, T. Pohlen, Z. Gong, D. Toyama, C. d. M. d’Autume, Y. Li, T. Terzi, V. Mikulik, I. Babuschkin, A. Clark, D. d. L. Casas, A. Guy, C. Jones, J. Bradbury, M. Johnson, B. Hechtman, L. Weidinger, I. Gabriel, W. Isaac, E. Lockhart, S. Osindero, L. Rimell, C. Dyer, O. Vinyals, K. Ayoub, J. Stanway, L. Bennett, D. Hassabis, K. Kavukcuoglu, and G. Irving. Scaling language models: Methods, analysis & insights from training gopher, 2021.
- [64] L. Reynolds and K. McDonell. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–7, 2021.
- [65] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models, 2021.
- [66] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models, 2021.
- [67] A. Rutherford. *ANOVA and ANCOVA: a GLM approach*. John Wiley & Sons, 2011.
- [68] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022.
- [69] K. Sakaguchi, R. L. Bras, C. Bhagavatula, and Y. Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.
- [70] V. Sanh, A. Webson, C. Raffel, S. Bach, L. Sutawika, Z. Alyafeai, A. Chaffin, A. Stiegler, A. Raja, M. Dey, et al. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.
- [71] T. L. Scao, A. Fan, C. Akiki, E. Pavlick, S. Ilić, D. Hesslow, R. Castagné, A. S. Luccioni, F. Yvon, M. Gallé, et al. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*, 2022.
- [72] T. L. Scao, A. Fan, C. Akiki, E.-J. Pavlick, S. Ilić, D. Hesslow, R. Castagné, A. S. Luccioni, F. Yvon, M. Gallé, J. Tow, A. M. Rush, S. R. Biderman, A. Webson, P. S. Ammanamanchi, T. Wang, B. Sagot, N. Muennighoff, A. V. del Moral, O. Ruwase, R. Bawden, S. Bekman, A. McMillan-Major, I. Beltagy, H. Nguyen, L. Saulnier, S. Tan, P. O. Suarez, V. Sanh, H. Laurenccon, Y. Jernite, J. Launay, M. Mitchell, C. Raffel, A. Gokaslan, A. Simhi, A. S. Etxabe, A. F. Aji, A. Alfassy, A. Rogers, A. K. Nitzav, C. Xu, C. Mou, C. C. Emezue, C. Klamm, C. Leong, D. A. van Strien, D. I. Adelani, D. R. Radev, E. G. Ponferrada, E. Levkovizh, E. Kim, E. B. Natan, F. D. Toni, G. Dupont, G. Kruszewski, G. Pistilli, H. ElSahar, H. Benyamina, H. T. Tran, I. Yu, I. Abdulmumin, I. Johnson, I. Gonzalez-Dios, J. de la Rosa, J. Chim, J. Dodge, J. Zhu, J. Chang, J. Frohberg, J. L. Tobing, J. Bhattacharjee, K. Almubarak, K. Chen, K. Lo, L. von Werra, L. Weber, L. Phan, L. B. Allal, L. Tanguy, M. Dey, M. R. Muñoz, M. Masoud, M. Grandury, M. vSavsko, M. Huang, M. Coavoux, M. Singh, M. T.-J. Jiang, M. C. Vu, M. A. Jauhar, M. Ghaleb, N. Subramani, N. Kassner, N. Khamis, O. Nguyen, O. Espejel, O. de Gibert, P. Villegas, P. Henderson, P. Colombo, P. A. Amuok, Q. Lhoest, R. Harliman, R. Bommasani, R. L’opez, R. Ribeiro, S. Osei, S. Pyysalo, S. Nagel, S. Bose, S. H. Muhammad, S. Sharma, S. Longpre, S. Nikpoor, S. Silberberg, S. Pai, S. Zink, T. T. Torrent, T. Schick, T. Thrush, V. Danchev, V. Nikoulina, V. Laippala, V. Lepercq, V. Prabhu, Z. Alyafeai, Z. Talat, A. Raja, B. Heinzerling, C. Si, E. Salesky, S. J. Mielke, W. Y. Lee, A. Sharma, A. Santilli, A. Chaffin, A. Stiegler, D. Datta, E. Szczechla, G. Chhablani, H. Wang, H. Pandey, H. Strobel, J. A. Fries, J. Rozen, L. Gao, L. Sutawika, M. S. Bari, M. S. Al-shaibani, M. Manica, N. V. Nayak, R. Teehan, S. Albanie, S. Shen, S. Ben-David, S. H. Bach, T. Kim, T. Bers, T. Février, T. Neeraj, U. Thakker, V. Raunak, X. Tang, Z. X. Yong, Z. Sun, S. Brody, Y. Uri, H. Tojarieh, A. Roberts, H. W. Chung, J. Tae, J. Phang, O. Press, C. Li, D. Narayanan, H. Bourfoune, J. Casper, J. Rasley, M. Ryabinin, M. Mishra, M. Zhang, M. Shoeybi, M. Peyrounette, N. Patry, N. Tazi, O. Sansevieri, P. von Platen, P. Cornette, P. F. Lavallée, R. Lacroix, S. Rajbhandari, S. Gandhi, S. Smith, S. Requena, S. Patil, T. Dettmers, A. Baruwa, A. Singh, A. Cheveleva, A.-L. Ligozat, A. Subramonian, A. N’ev’el, C. Lovering, D. H.

- Garrette, D. R. Tunuguntla, E. Reiter, E. Taktasheva, E. Voloshina, E. Bogdanov, G. I. Winata, H. Schoelkopf, J.-C. Kalo, J. Novikova, J. Z. Forde, X. Tang, J. Kasai, K. Kawamura, L. Hazan, M. Carpuat, M. Clinciu, N. Kim, N. Cheng, O. Serikov, O. Antverg, O. van der Wal, R. Zhang, R. Zhang, S. Gehrmann, S. Mirkin, S. O. Pais, T. Shavrina, T. Scialom, T. Yun, T. Limisiewicz, V. Rieser, V. Protasov, V. Mikhailov, Y. Pruksachatkun, Y. Belinkov, Z. Bamberger, Z. Kasner, A. Rueda, A. Pestana, A. Feizpour, A. Khan, A. Faranak, A. S. R. Santos, A. Hevia, A. Unldreaj, A. Aghagol, A. Abdollahi, A. Tammour, A. HajiHosseini, B. Behroozi, B. O. Ajibade, B. K. Saxena, C. M. Ferrandis, D. Contractor, D. M. Lansky, D. David, D. Kiela, D. A. Nguyen, E. Tan, E. Baylor, E. Ozoani, F. T. Mirza, F. Ononiwu, H. Rezanejad, H. Jones, I. Bhattacharya, I. Solaiman, I. Sedenko, I. Nejadgholi, J. Passmore, J. Seltzer, J. B. Sanz, K. Fort, L. M. Dutra, M. Samagaio, M. Elbadri, M. Mieskes, M. Gerchick, M. Akinlolu, M. McKenna, M. Qiu, M. K. K. Ghauri, M. Burynok, N. Abrar, N. Rajani, N. Elkott, N. Fahmy, O. Samuel, R. An, R. P. Kromann, R. Hao, S. Alizadeh, S. Shubber, S. L. Wang, S. Roy, S. Viguer, T.-C. Le, T. Oyebade, T. N. H. Le, Y. Yang, Z. K. Nguyen, A. R. Kashyap, A. Palasciano, A. Callahan, A. Shukla, A. Miranda-Escalada, A. K. Singh, B. Beilharz, B. Wang, C. M. F. de Brito, C. Zhou, C. Jain, C. Xu, C. Fourrier, D. L. Perin'an, D. Molano, D. Yu, E. Manjavacas, F. Barth, F. Fuhrmann, G. Altay, G. Bayrak, G. Burns, H. U. Vrabec, I. I. Bello, I. Dash, J. S. Kang, J. Giorgi, J. Golde, J. D. Posada, K. Sivaraman, L. Bulchandani, L. Liu, L. Shinzato, M. H. de Bykhovetz, M. Takeuchi, M. Pàmies, M. A. Castillo, M. Nezhurina, M. Sanger, M. Samwald, M. Cullan, M. Weinberg, M. Wolf, M. Mihaljcic, M. Liu, M. Freidank, M. Kang, N. Seelam, N. Dahlberg, N. M. Broad, N. Muellner, P. Fung, P. Haller, R. Chandrasekhar, R. Eisenberg, R. Martin, R. L. Canalli, R. Su, R. Su, S. Cahyawijaya, S. Garda, S. S. Deshmukh, S. Mishra, S. Kiblawi, S. Ott, S. Sang-aroonisri, S. Kumar, S. Schweter, S. P. Bharati, T. A. Laud, T. Gigant, T. Kainuma, W. Kusa, Y. Labrak, Y. Bajaj, Y. Venkatraman, Y. Xu, Y. Xu, Y. chao Xu, Z. X. Tan, Z. Xie, Z. Ye, M. Bras, Y. Belkada, and T. Wolf. Bloom: A 176b-parameter open-access multilingual language model. *ArXiv*, abs/2211.05100, 2022.
- [73] W. Shi, X. Han, M. Lewis, Y. Tsvetkov, L. Zettlemoyer, and S. W.-t. Yih. Trusting your evidence: Hallucinate less with context-aware decoding. *arXiv preprint arXiv:2305.14739*, 2023.
- [74] I. Solaiman, M. Brundage, J. Clark, A. Askell, A. Herbert-Voss, J. Wu, A. Radford, G. Krueger, J. W. Kim, S. Kreps, et al. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*, 2019.
- [75] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*.
- [76] A. Spangher, X. Hua, Y. Ming, and N. Peng. Sequentially controlled text generation. *arXiv preprint arXiv:2301.02299*, 2023.
- [77] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto. Stanford alpaca: An instruction-following llama model. [https://github.com/tatsu-lab/stanford\\_alpaca](https://github.com/tatsu-lab/stanford_alpaca), 2023.
- [78] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- [79] B. Wang and A. Komatsuzaki. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. <https://github.com/kingoflolz/mesh-transformer-jax>, May 2021.
- [80] X. Wang, J. Wei, D. Schuurmans, Q. V. Le, E. H. Chi, S. Narang, A. Chowdhery, and D. Zhou. Self-consistency improves chain of thought reasoning in language models. In *ICLR 2023*, 2023.
- [81] J. Wei, M. Bosma, V. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.
- [82] J. Wei, X. Wang, D. Schuurmans, M. Bosma, b. ichter, F. Xia, E. Chi, Q. V. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc., 2022.
- [83] C. Xu, Q. Sun, K. Zheng, X. Geng, P. Zhao, J. Feng, C. Tao, and D. Jiang. Wizardlm: Empowering large language models to follow complex instructions, 2023.
- [84] K. Yang and D. Klein. Fudge: Controlled text generation with future discriminators. *arXiv preprint arXiv:2104.05218*, 2021.
- [85] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

# Appendix

## Table of Contents

---

<b>A Author Contributions</b>	<b>16</b>
<b>B Additional Related Works</b>	<b>17</b>
B.1 CFG . . . . .	17
B.2 Generative Guidance in NLP . . . . .	17
<b>C Charts</b>	<b>17</b>
C.1 General benchmarks . . . . .	17
C.2 Accuracy vs. FLOP . . . . .	18
C.3 HumanEval benchmark . . . . .	21
C.4 Deliberative Prompting: Chain-of-Thought . . . . .	24
<b>D Additional experiments</b>	<b>28</b>
D.1 Machine translation . . . . .	28
D.2 Prompting experiments for code generations . . . . .	28
<b>E Generation samples</b>	<b>30</b>
E.1 Continuations . . . . .	30
<b>F Further Comparison between CFG and Instruction-Tuning</b>	<b>34</b>
<b>G Experiments with GPT4All</b>	<b>34</b>
G.1 System prompts . . . . .	34
G.2 User prompts . . . . .	35

---

## A Author Contributions

This work is a spontaneous collaboration between EleutherAI members and EleutherAI’s Discord’s members.

**Guillaume Sanchez** came up with the initial theory, code and preliminary experiments, then reached EleutherAI in search for collaborators. He wrote the code for 2 and associated figures, redacted Sections 2.1, 2.2. He wrote the code and ran the GPT-J experiment mentioned in 3.3.1. He built the platform for the human experiment , publicized the experiment to get votes, and compiled the results for 3.4.

**Honglu Fan** proofread 2.2, 2.1, redacted Section 3’s introduction and 3.1, Appendix C.1C.2, C.3. Designed and ran the experiments for Section 3.3. He took care of running the experiments of Section 3.1 thanks to his access to CoreWeave and Stability’s computing cluster.

**Alexander Spangher** proofread the paper and is the primary writer/editor and redactor of it. He wrote the Introduction, Section 2’s introduction, Section 4, Section 5’s introduction, Appendix B and the Conclusion, regenerated many of the figures, and proofread everything.

He designed, ran and redacted the experiments in Sections 5.1, 5.2, and Appendix F.

**Elad Levi** designed and ran the Chain-Of-Thoughts experiments in Section 3.2. He wrote a preliminary version of Sections 2.1, 2.2 and redacted Section 3.2 and Appendix C.4.

**Pawan Ammanamanchi** designed, ran and redacted the machine translation experiments of Appendix D.1.

**Stella Biderman** supervised the process. She proofread the paper, suggested the experiments to run in 3.1 and how to run them with EleutherAI’s LM Harness. She suggested the GPT-J code generation experiment of section 3.3.1.

## B Additional Related Works

### B.1 CFG

The work on CFG is based on Classifier Guided Diffusion [28], which demonstrates that  $\gamma$  allows for trading fidelity and diversity. Artists using Stable Diffusion, an open-source product built on [66], commonly believe that effective prompt engineering and creative pictures require strong prompt conditioning happening for  $\gamma > 1$ . This belief is supported by experiments, such as those conducted with Imagen [68], which show that the prompt correlates more with the image as  $\gamma$  increases.

### B.2 Generative Guidance in NLP

Co-temporaneously with the earliest advances in neural language modeling [35] came the recognition that the outputs of these models had to be guided in order to be coherent [7] and focused [38]. And when larger, higher-performing models like GPT [62, 15] began to show real-world use-cases, the recognition emerged of the need to control their output [74] to guard against toxic content [34] and bias [30].

A central thrust in recent NLP research been to address the above concerns, and approaches have been targeted at nearly every step of training and querying models, from dataset curation [2] and training [40], to response-alignment [57] and prompt-identification [34].

Our work aligns with efforts to control the output of language models by controlling the model’s outputted vocabulary distribution  $p(x_n|x_{<n})$ . Early efforts in this vein aimed at increasing coherence include now-standard techniques like temperature-scaling [17], nucleus sampling [38] and heuristics (e.g. repetition penalties [31]).

In parallel, more sophisticated approaches to control the output of language models by moderating the vocabulary distribution emerged within the line of “controlled text generation”. Works in this vein emerged *after* the earliest attempt at controlled-generation, CTRL [40], where researchers pretrained a language model to be aware of prompts as well as “control codes”,  $a$  that could produce conditional generations,  $p(x_n|x_{<n}, a)$ , (where  $a \in \{ \text{“Science”}, \text{“Romance”}, \text{“Mystery”}\dots \}$ ) that could produce conditional generations, steer the prompt continuation away from the initial generation. This work established the idea of “controlled generation”; it was quickly followed by the Plug and Play Language model (PPLM) [24]. PPLM was the earliest work achieving controlled generation through moderating the vocabulary distribution of a vanilla pretrained language model. Authors used Bayes Rule to factorize the conditional distribution  $p(x_n|x_{<n}, a) \propto p(x_n|x_{<n})p(a|x_n, x_{<n})$ . Other works followed in this vein [43, 84, 76, 50, 44]. Authors used a naive pretrained language model like GPT2 [62] to model  $p(x_n|x_{<n})$  and trained a discriminator  $p(a|x)$  on labeled datasets, and then added together the two log probabilities to obtain the controlled distribution.

Efforts at controlled generation largely fell out of favor with the advent of instruction-tuning [57]; using instruction-tuned models like GPT3 [15], users could simply the model to “write happy text”, or “write very happy text”. However, experiments with moderating the vocabulary distribution continued, and researchers recently showed that combining two models – an expert model and a weak model – could produce more fluent text [45]. In this paper, instead of our CFG formulation ( $\lambda \log p(x|y) - (1 - \lambda) \log p(x)$ ), authors used two models, a weak model  $f_w$  and a strong model  $f_s$ , to do:  $f_s(x|y) - f_w(x|y)$  in order to generate more inventive, creative language that was even *more* in the direction of  $f_s$  than would have been.

## C Charts

In this section, we collect some charts that visualize results in Section 3.1, 3.3 and 5.

### C.1 General benchmarks

In Section 3.1, GPT-2, Pythia, LLaMA model families are analyzed with and without CFG. In addition to Table 2, we make plots of each model family with x-axis being the CFG strength and the y-axis being the accuracy. It aims to provide a more direct view of how model size affect the accuracy-to- $\gamma$  curves while scaling in the same model family. The plots are shown in Figure 8, 9 and 10.

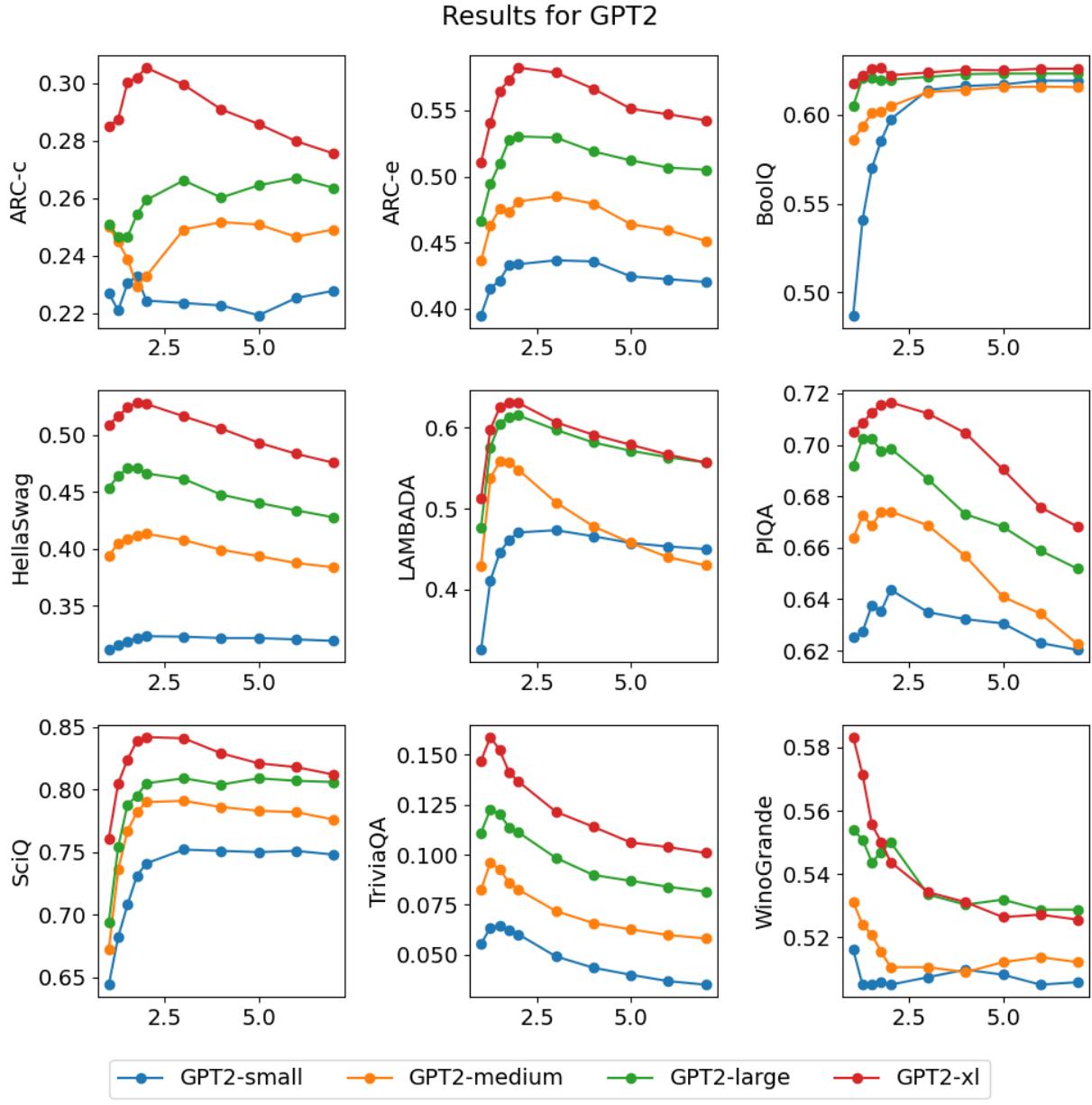


Figure 8: Standard benchmarks over various CFG strengths for GPT2 models

We run TriviaQA based on the LLaMA [78] methodology, however we perform substring match rather than exact match. This stems from manual analysis which showed that exact matching disqualified answers like "Mark Twain" (with quotes) or His name is Mark Twain instead of the exact Mark Twain.

## C.2 Accuracy vs. FLOP

In Section 4, we present the finding that a model using CFG can generally perform as well as a model twice as large without CFG. The detailed charts are presented in this subsection.

Results for Pythia

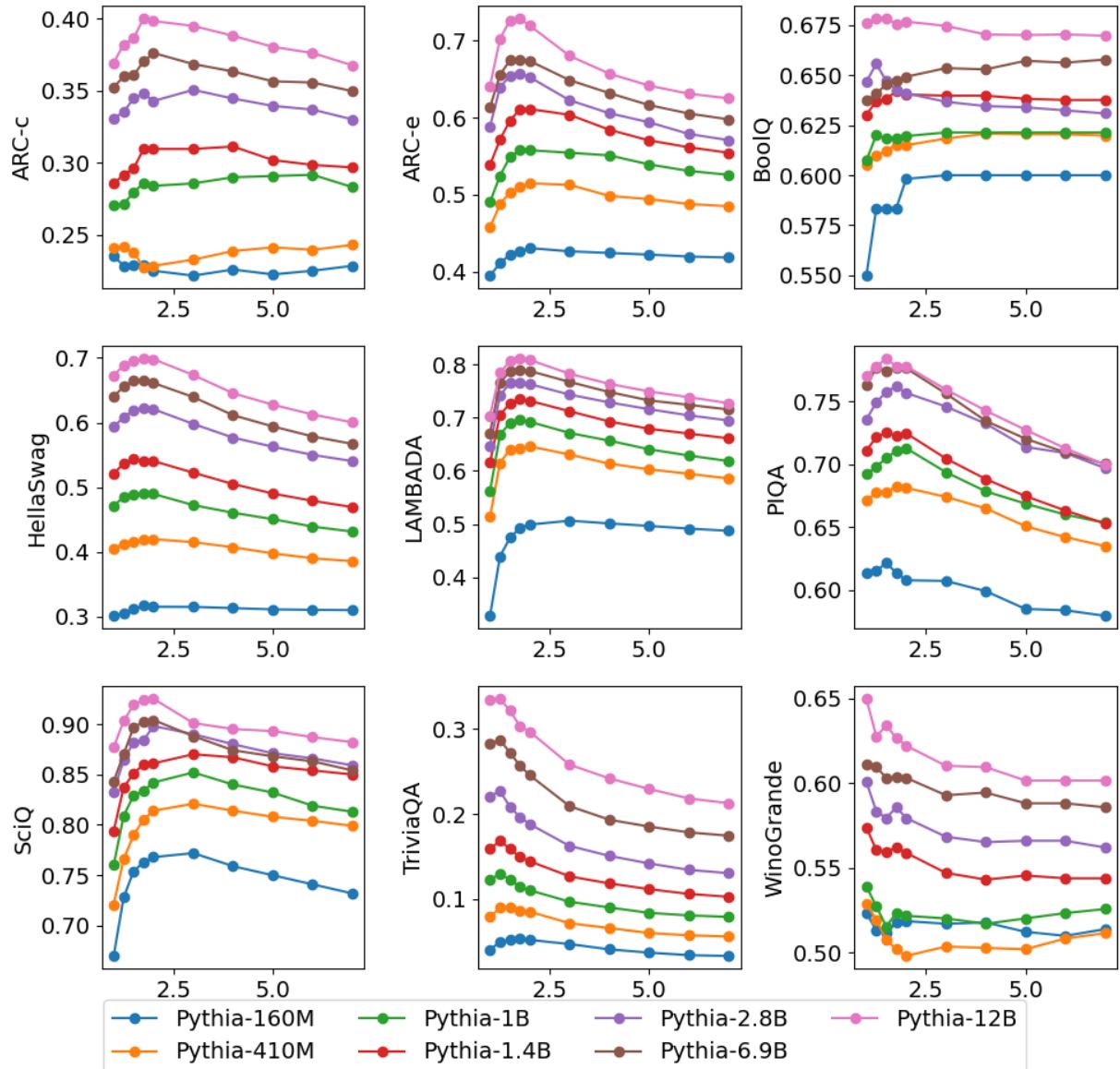


Figure 9: Standard benchmarks over various CFG strengths for Pythia models

Results for LLaMA

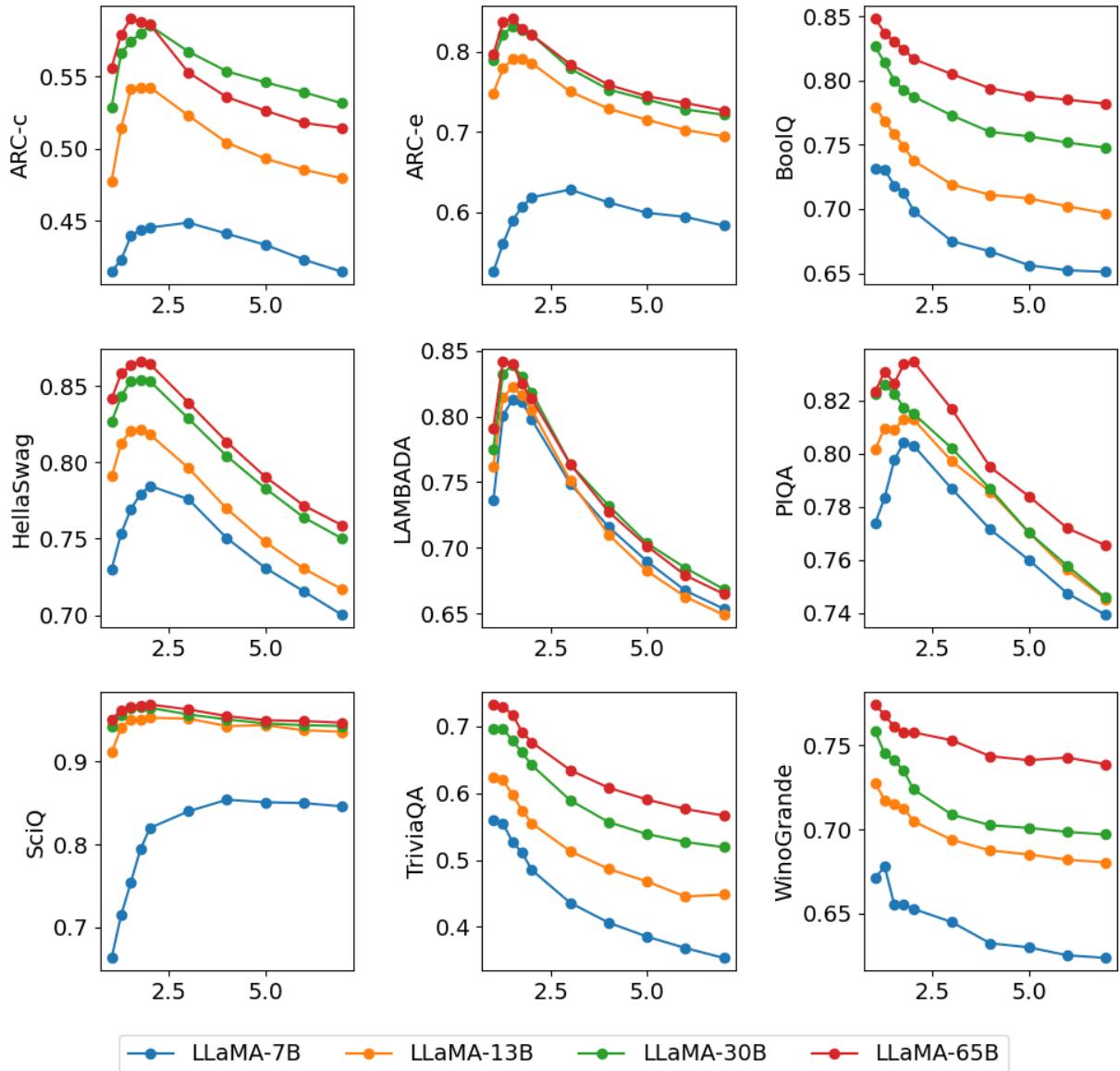


Figure 10: Standard benchmarks over various CFG strengths for LLaMA models

	<i>p</i> -value	Win
Lambada	0.000	CFG
WinoGrande	0.003	Vanilla
SciQ	0.008	CFG
TriviaQA	0.008	Vanilla
HellaSwag	0.012	$p > .01$
PiQA	0.030	$p > .01$
ARC-c	0.216	$p > .01$
BoolQ	0.345	$p > .01$
ARC-e	0.355	$p > .01$

Table 4: ANCOVA *p*-value results for plots shown in Figure 11. We calculate ANCOVA on log-transformed variables and calculate significance at  $p = .01$ .

$\gamma$	temperature = 0.2			temperature = 0.6			temperature = 0.8		
	k=1	k=10	k=100	k=1	k=10	k=100	k=1	k=10	k=100
1.0	11.0%	17.0%	<b>22.0%</b>	8.9%	18.2%	23.7%	7.2%	17.2%	<b>29.4%</b>
1.1	<b>11.8%</b>	<b>18.1%</b>	20.1%	<b>10.0%</b>	<b>19.7%</b>	<b>25.5%</b>	7.8%	17.1%	22.5%
1.25	11.4%	17.3%	18.9%	9.7%	18.4%	23.7%	<b>8.3%</b>	<b>18.2%</b>	24.9%
1.5	10.9%	16.7%	18.3%	9.9%	19.3%	24.9%	8.0%	18.0%	26.1%
1.75	10.3%	16.0%	18.2%	9.2%	18.3%	23.7%	7.7%	16.9%	24.2%
2.0	8.6%	14.6%	17.6%	7.6%	16.6%	20.1%	7.4%	16.5%	21.3%

Table 5: CodeGen-350M-mono results

With the same data points as Section C.1, we reorganize them into inference accuracy vs. FLOP<sup>8</sup> per token plots so that we can compare the performance of a model with CFG (doubled inference FLOP) and a model without CFG but twice as big. We show all the plots in Figure 11.

Note that:

1. **The location of each data point in the charts ignores the model size and only reflects its inference FLOP per token.** For example, a 1.4B model with CFG (doubled inference FLOP) will show up near a 2.8B model without CFG if they perform closely, despite the fact that such 1.4B model is more useful in practice due to the saving on training and VRAM.
2. **The data points in the charts only reflect the inference cost and ignoring the training cost.** For example, when a 1.4B model gets boosted to the accuracy of a 2.8B model by using CFG, the inference costs are similar but to train a 1.4B model takes less compute.

For Lambada and SciQ, CFG is a clear winner which improves the whole compute-accuracy curve while for WinoGrande, CFG impacts negatively. The rest are mixed.

This entails that for the same inference cost, CFG can emulate a model that has twice the parameter count. This drastically reduces the VRAM usage needed to run the models which is the current bottleneck, and reduces the training cost. To further justify this, Table 11 is a breakdown of the ANCOVA *p*-values for each chart between the regression line of the CFG group (in red) and the one of the vanilla group (in blue). We choose the *p*-value cutoff at 0.01 according to [67], and higher than 0.01 means an insignificant difference between the regression lines of the two groups.

### C.3 HumanEval benchmark

In Section 3.3.1, we explain our experiments on CodeGen-350M-mono, CodeGen-2B-mono and CodeGen-6B-mono and show their performances in the HumanEval benchmark with various CFG for temperature 0.2 in Table 2. The full results for temperature = 0.2, 0.6, 0.8 are shown below in Table 5, 6 and 7). We also put the pass@*k*-to- $\gamma$  curves of different temperatures together to show how the temperatures affect the impact of CFG when the model size and *k* are fixed in Figure 12, 13 and 14.

<sup>8</sup>FLOP: floating point operations

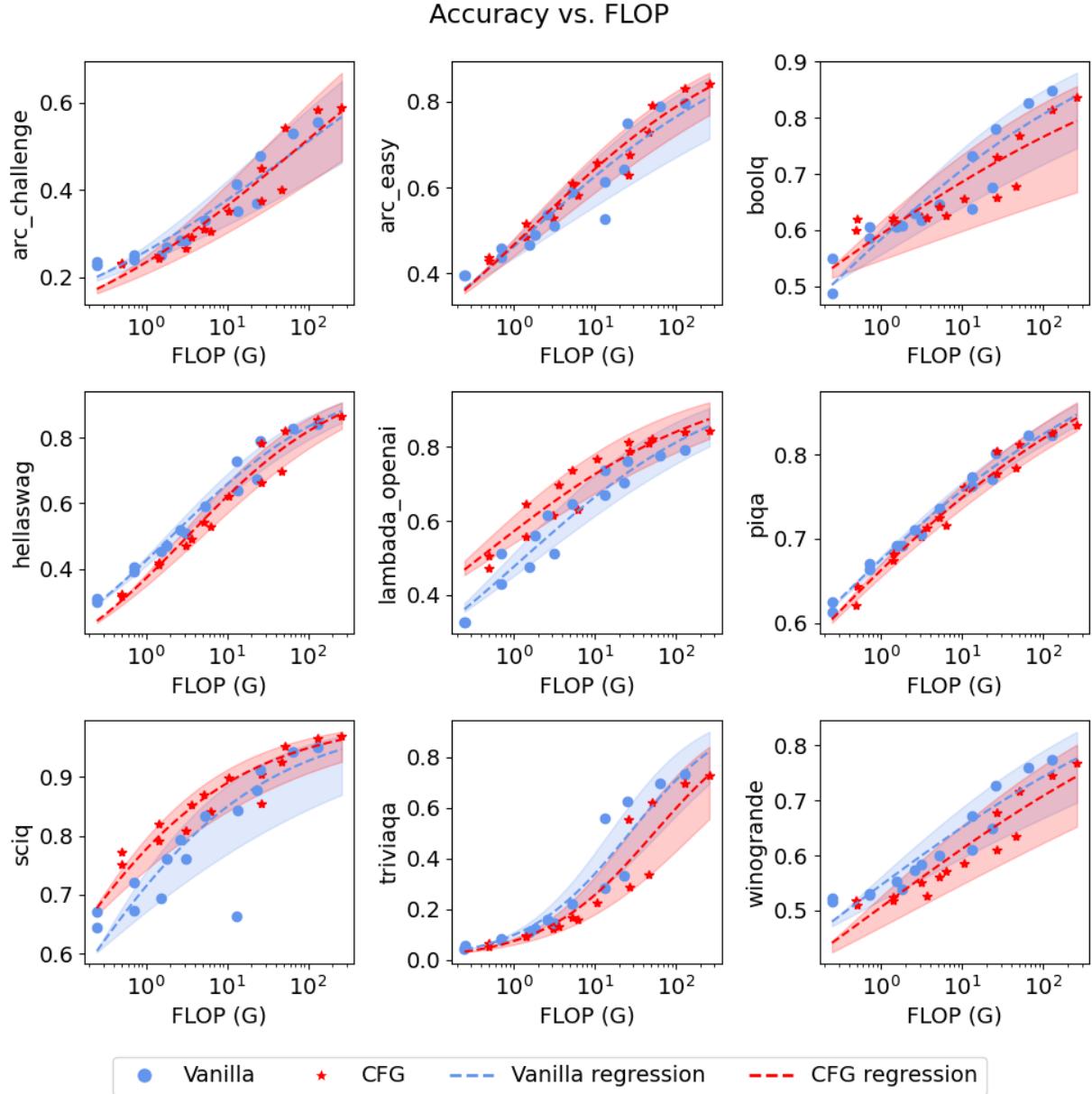


Figure 11: Accuracy vs. FLOP per token at inference.

**Blue point:** a model without CFG from any of the three model families (GPT-2, Pythia, LLaMA).

**Red point:** a model with the best CFG from any of the three model families.

**The dashed curves:** the regression curves (logistic regression between log-FLOP and accuracy) of their groups.

$\gamma$	temperature = 0.2			temperature = 0.6			temperature = 0.8		
	k=1	k=10	k=100	k=1	k=10	k=100	k=1	k=10	k=100
1.0	19.5%	25.5%	<b>29.8%</b>	15.9%	29.3%	36.5%	12.3%	26.4%	33.5%
1.1	20.4%	25.4%	28.0%	16.3%	29.3%	36.5%	13.8%	29.0%	<b>38.3%</b>
1.25	19.7%	25.4%	28.0%	17.4%	30.1%	38.3%	14.1%	28.7%	37.6%
1.5	<b>20.9%</b>	<b>26.7%</b>	29.2%	<b>18.3%</b>	<b>31.7%</b>	<b>40.1%</b>	<b>14.9%</b>	<b>29.1%</b>	36.5%
1.75	20.4%	26.2%	28.6%	17.7%	30.4%	35.9%	14.3%	28.3%	34.1%
2.0	16.5%	22.4%	24.4%	13.7%	25.2%	32.2%	11.3%	23.9%	31.6%

Table 6: CodeGen-2B-mono results

$\gamma$	temperature = 0.2			temperature = 0.6			temperature = 0.8		
	k=1	k=10	k=100	k=1	k=10	k=100	k=1	k=10	k=100
1.0	19.5%	25.5%	<b>29.8%</b>	15.9%	29.3%	36.5%	12.3%	26.4%	33.5%
1.1	20.4%	25.4%	28.0%	16.3%	29.3%	36.5%	13.8%	29.0%	<b>38.3%</b>
1.25	19.7%	25.4%	28.0%	17.4%	30.1%	38.3%	14.1%	28.7%	37.6%
1.5	<b>20.9%</b>	<b>26.7%</b>	29.2%	<b>18.3%</b>	<b>31.7%</b>	<b>40.1%</b>	<b>14.9%</b>	<b>29.1%</b>	36.5%
1.75	20.4%	26.2%	28.6%	17.7%	30.4%	35.9%	14.3%	28.3%	34.1%
2.0	16.5%	22.4%	24.4%	13.7%	25.2%	32.2%	11.3%	23.9%	31.6%

Table 7: CodeGen-6B-mono results

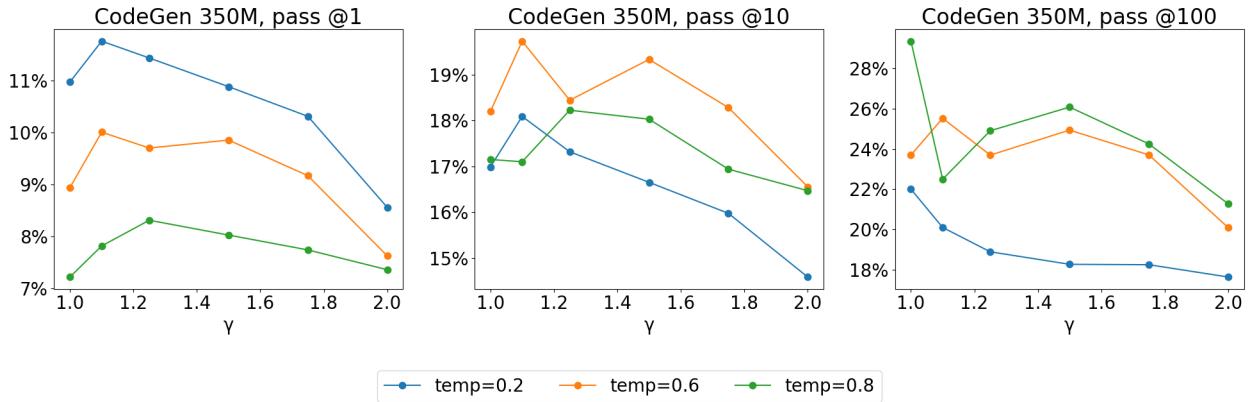


Figure 12: CodeGen-350M-mono performance on HumanEval with various CFG strengths

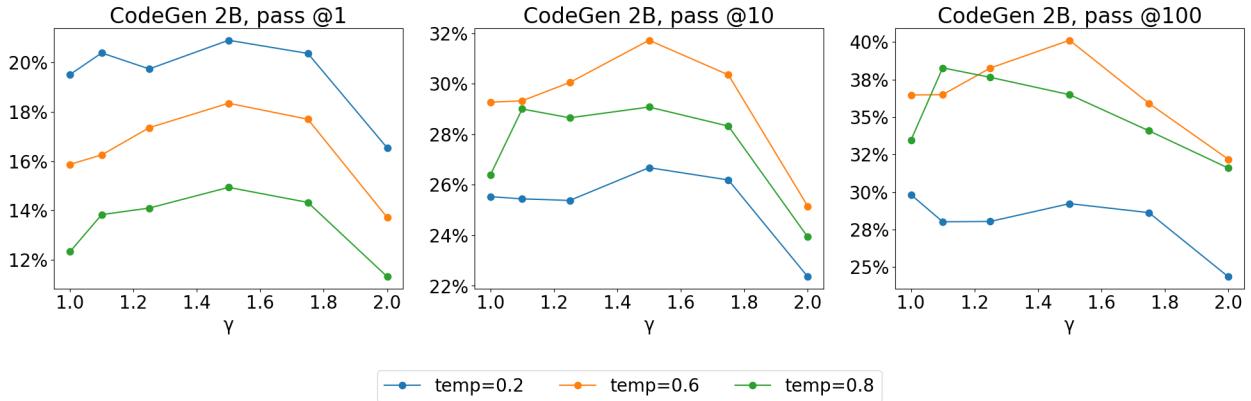


Figure 13: CodeGen-2B-mono performance on HumanEval with various CFG strengths

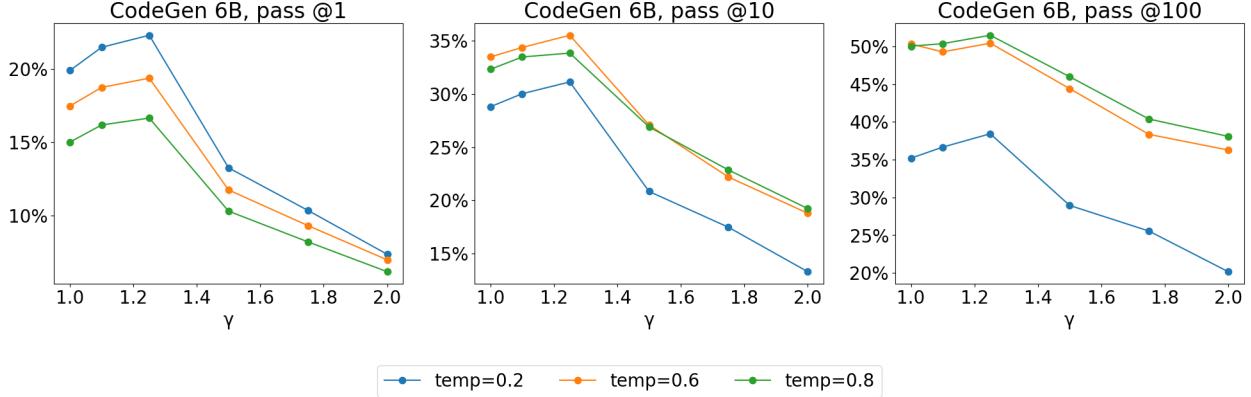


Figure 14: CodeGen-6B-mono performance on HumanEval with various CFG strengths

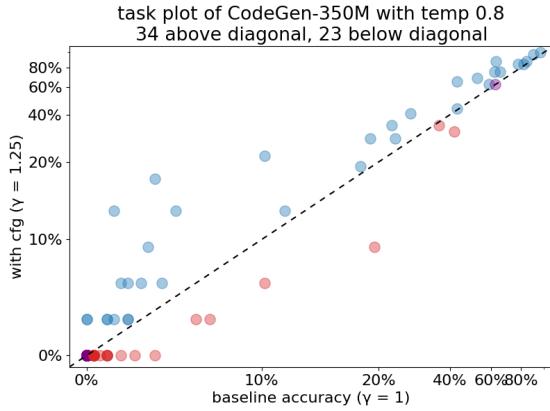
P3 Dataset	mean	std	count
<b>Highest ⟨ CFG, Instruct⟩ Similarities</b>			
SuperGLUE wsc.fixed p is are r score eval	31.89	+/-22.06	42
SciQ Multiple Choice Closed Book	5.82	+/-13.27	43
CosE v1.11 description question option text	5.70	+/-9.05	43
RottenTomatoes WriterExpressed Sentiment	4.93	+/-7.45	41
WinograndeXL fill in the blank	4.42	+/-10.51	44
RottenTomatoes TextExpressed Sentiment	2.93	+/-7.98	45
Quarel: choose between	2.51	+/-12.39	43
SuperGLUE wic GPT 3 prompt score eval	2.15	+/-5.94	44
WinograndeDebiased Replace score eval	2.02	+/-24.46	41
PAWS final context question (no label)	1.37	+/-4.81	43
<b>Lowest ⟨ CFG, Instruct⟩ Similarities</b>			
paws labeled final paraphrase task	-11.71	+/-11.03	42
super glue copa more likely	-11.94	+/-6.38	45
piqa Does this solution make sense sol2	-12.22	+/-9.24	42
super glue copa cause effect score eval	-12.82	+/-5.8	41
rotten tomatoes Sentiment with choices	-13.07	+/-7.96	41
super glue copa plausible alternatives score eval	-15.07	+/-5.69	41
super glue copa C1 or C2 premise so because	-15.38	+/-6.43	41
super glue copa more likely score eval	-16.54	+/-5.45	43
cos e v1.11 question option description id	-17.60	+/-14.06	41
rotten tomatoes Reviewer Enjoyment Yes No	-18.16	+/-16.02	45

Table 8: Datasets in P3 where Instruction-Tuned models were the most and least similar, in terms of top-p overlap, to CFG models. The count column shows the number of datapoints that were sampled from each dataset to calculate the overlap.

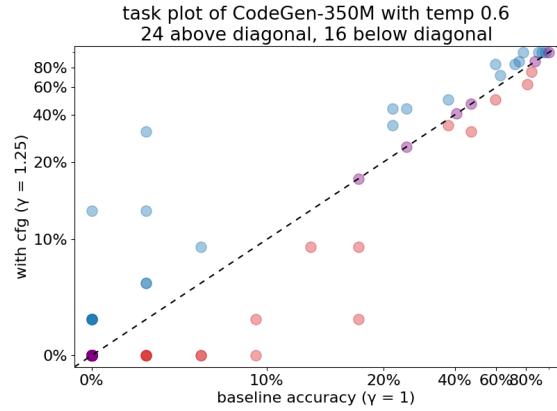
In addition, we breakdown the result of CodeGen-350M-mono on HumanEval benchmark into individual tasks. We plot the “accuracy with cfg” vs. “accuracy without cfg” charts to visualize the outperform/underperform distributions among all tasks. The plots are shown in Figure 15c, 15b and 15a.

#### C.4 Deliberative Prompting: Chain-of-Thought

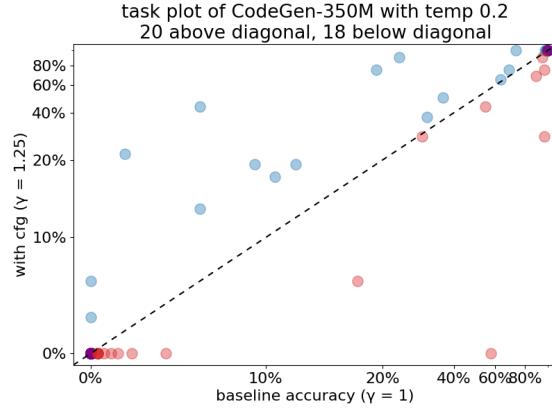
In this subsection we provide additional results for 3.2. In Figure 15 we provide results on AQuA dataset and in Tables 15 and 14 we provide a qualitative comparison of CoT with and without CFG. These results support our finding that using CFG increases the percentage of CoT which results in a valid answer and boost the model performances.



(a) CodeGen-350M-mono HumanEval task-by-task plot with  
temp=0.8  
**Blue:** CFG outperforms,  
**Purple:** CFG ties with the baseline,  
**Red:** CFG underperforms



(b) CodeGen-350M-mono HumanEval task-by-task plot with  
temp=0.6  
**Blue:** CFG outperforms,  
**Purple:** CFG ties with the baseline,  
**Red:** CFG underperforms



(c) CodeGen-350M-mono HumanEval task-by-task plot with  
temp=0.2  
**Blue:** CFG outperforms,  
**Purple:** CFG ties with the baseline,  
**Red:** CFG underperforms

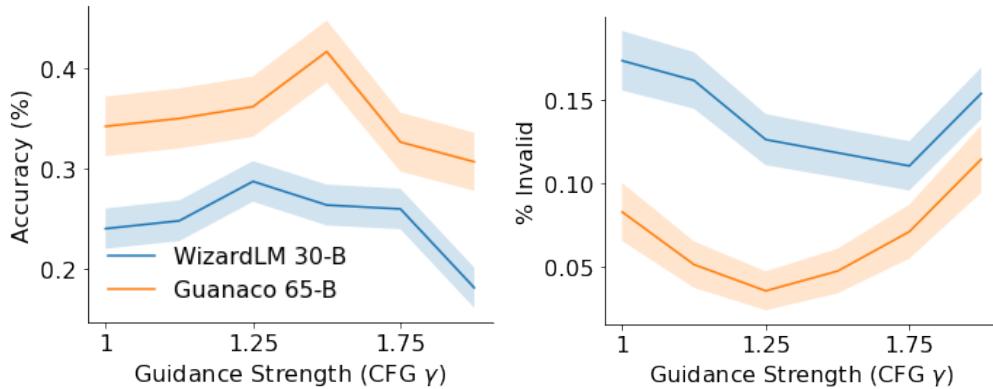


Figure 15: CFG impact on chain-of-thought prompting with respect to AQuA dataset. For small CFG values, using CFG increases the percentage of chains which end in a valid answer structure while increasing the model accuracy. For large values the invalid percentage remains small but the accuracy drop.

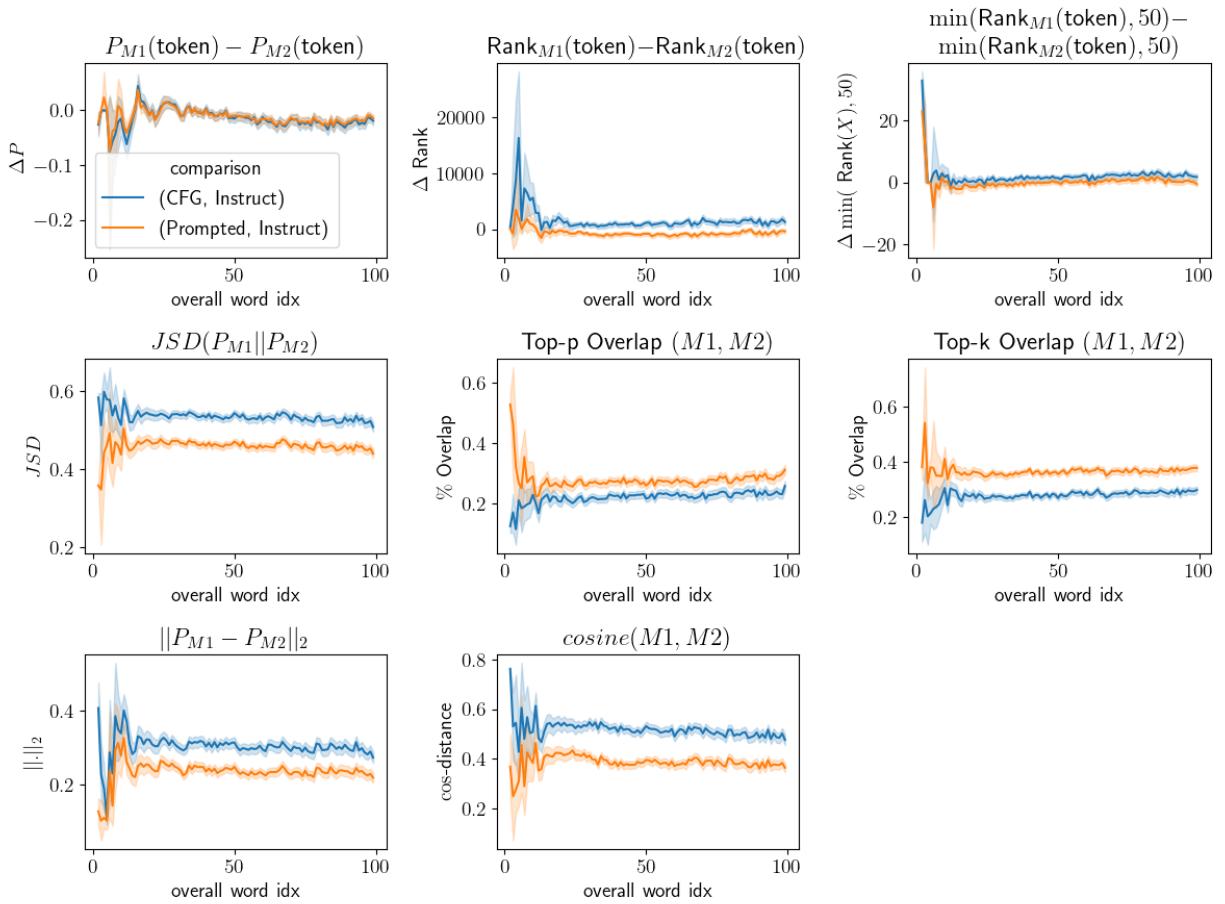


Figure 16: Comparison of (CFG- $\gamma = 1.5$ , Instruct) logits across a large sample set from P3.

---

### Top Sentences in P3 where CFG is MOST Similar to Instruction-Tuned Models

---

Build a movie plot around this: What is the team? Rag-tag bunch of girls  
Here's a complex question that requires someone to reason about the input, can you answer it? What city was the capital of the Ostrogothic Kingdom and the birth place of Ornella Fiorentini?  
Who had more of their English novels turned into Oscar-nominated films, Raja Rao or Pat Conroy?  
Nokia, Texas Instruments and other leading makers of mobile phones have formally complained to Brussels that Qualcomm, the US mobile chipmaker, has unfairly used its patents on 3G technologies. Question: Texas Instruments produces mobile phones. True or False?  
Context: Patting her back, the woman smiled at the girl . Question: "her" is the woman. True or false? Answer:  
Take the following as truth: The American Combat Association is a small mixed martial arts company founded by Olympic wrestler, world Abu Dhabi champion and UFC fighter Kamal Shalorus and professional mixed martial arts fighter, Broadcaster and American professional wrestler Matthew "The Ganimal" Granahan. Then the following statement: "The American Combat Association was founded by two Olympic wrestlers." is true, false, or inconclusive?  
Pick the most correct option to answer the following question. Some antibiotics used to treat infections in humans are also used to treat chickens, but some groups oppose this practice. The overuse of the antibiotics will most likely influence the natural selection of which type of organisms? Options: - A: chickens that naturally make the antibiotics - B: microbes that are resistant to the antibiotics - C: microbes that are susceptible to the antibiotics - D: chickens that are resistant to infection  
Jennifer dragged Felicia along to a self help workshop about how to succeed, because \_ wanted some company.  
Replace the \_ in the above sentence with the correct option: - Jennifer - Felicia  
Brian could learn to swim with the right instruction, but it was hard to tell whether lifeguard Matthew was qualified to provide it, since \_ had never swum before. Replace the \_ in the above sentence with the correct option: - Brian - Matthew

---

Table 9: Top sentences in P3 where CFG is similar to Instruction-Tuned models, as measured by top- $p$  overlap.

---

### Sentences in P3 where CFG is LEAST Similar to Instruction-Tuned Models

---

How do you feel about your current weight and eating habits ?  
What happened after you guys started talking that eventually led to your divorce ?  
Given a goal and a wrong solution, rewrite it to give a correct solution. Goal: how do you train a puppy? Solution:  
Corrected solution:  
What might have happened since I was a democrat in my first year ?  
What do you usually do when you meet a guy for the first time ?  
What did you do that caused you to be in the bathroom all day ?  
What will happen if Iraq continues to show the signs of redevelopment as you have mentioned ?  
What might happen if we show our true selves to the people we love ?  
I would like to create a garden on my balcony. What is the first thing I should do?  
What will you do if a branch falls off one of the oaks ?  
What will you do now that you define as taking action ?  
The abode of the Greek gods was on the summit of Mount Olympus, in Thessaly. Question: Mount Olympus is in Thessaly. True or False?  
Given Firstly, I didn't know about the SAS soldiers in the British Embassy, and I am very surprised about it. Very surprised indeed, Ambassador. Secondly I do not think it is a good idea to attack a plane with a hundred and seven passengers in it and "take it apart" as you say. Is it guaranteed true that "it is a good idea to attack a plane with a hundred and seven passengers in it and 'take it apart'"? Yes, no, or maybe?  
'Cote d'Ivoire's President, Laurent Gbagbo, promulgated new election laws on July 14. Question: President Laurent Gbagbo lives in Cote d'Ivoire. True or False?  
'the real star of this movie is the score , as in the songs translate well to film , and it's really well directed . The sentiment expressed for the movie is '  
My closet was messy. so... Choose between: - I organized it. - I decorated it.

---

Table 10: Sentences in P3 where CFG is LEAST similar to Instruction-Tuned models, as measured by top- $p$  overlap.

## D Additional experiments

### D.1 Machine translation

We evaluate using Classifier-Free Guidance for machine translation on a variety of models. We choose the WMT14 fr-en [13] as the dataset of choice to understand if CFG would also help multilingual datasets. We run 0-shot experiments on Bloom-3B [72], a multilingual model trained on 49 languages. We also test on RedPajama-Incite-Base-3B, trained on 1.5T tokens of English text and mT0 [52] a prompt tuned sequence-to-sequence model. For the Bloom-3B model, we test for multiple prompts and perform 1-shot experiments as well. All scores are measured in BLEU.

We find that for this generation task,  $\gamma$  ranging between 1.1 to 1.25 yield the best results and perform increasingly worse at higher values. We additionally observe that the method is prompt-invariant, showing gains regardless of the prompt choice in 0-shot performance. We do not see any improvements in the case of 1-shot performance for Bloom-3B. We also do not see any significant performance gains in the case of mT0, suggesting that prompt-tuned models might already be at the pinnacle of possible 0-shot performance.

### D.2 Prompting experiments for code generations

We summarize two exploratory experiments which are briefly mentioned in 3.3.1 and precedes our systematic evaluations on HumanEval.

1. The first experiment is to prompt GPT-J [79]<sup>9</sup> for code completions of certain languages, and analyze the consistencies between the prompt languages and the completion languages.
2. The second experiment is to prompt CodeGen-350M-mono [54] to complete a specific image generation function, and analyze multiple aspects of the completions (syntax, the return type, the return shape and the return quality).

Prompting GPT-J for different coding language is inspired by one of the experiments in [10]. Their observation is that the model often generates non-code or not the programming language it was prompted for.

We generate 100 samples (5 runs for 5 prompts) for each guidance strength  $\gamma = 1, 1.25, 1.5, 1.75$ . We observe the  $\gamma = 1$  baseline generating the correct programming language 73% of the time, jumping to 86% with  $\gamma = 1.25$  ( $p$ -value 0.01). See 12 for more details.

Next, we turn to CodeGen-350M-mono ([54]) for code completion for a fixed image generation function. The prompt is the following:

```
# Return a red square on a 32x32 picture in the form of numpy array with RGB channels
def draw() -> np.ndarray:
```

We produce 1600 completions for each CFG strength  $\gamma = 1.0, 2.0$ . The results are evaluated based on:

- syntax correctness (executing without errors),
- return type correctness (returning a numpy array),
- return shape correctness (having shape (32, 32, 3)),
- the  $l^2$ -distance to a reference picture (picture of pure color in red).

When calculating the  $l^2$ -distance, all pixels are normalized to the range [0, 1]. The result is summarized in Table 13.

The difference is fairly noticeable, where the biggest improvement comes from the return type correctness.

---

<sup>9</sup>GPT-J is not specifically trained for code generation task. But it was exposed to some code data in its training.

Model	$\gamma = 1$	$\gamma = 1.10$	$\gamma = 1.25$
Bloom-3B	14.16	<b>15.81</b>	14.16
RedPajama-Incite-3B	15.04	<b>17.24</b>	17.78
	$\gamma = 1$	$\gamma = 1.05$	$\gamma = 1.10$
Bloom-3B 1-shot	<b>29.84</b>	29.19	28.53
mt0	<b>29.77</b>	29.41	27.79

Table 11: BLEU scores for different  $\gamma$  for machine translation tasks. In the case of 1-shot and mt0, we experiment with  $\gamma$  values between 1 and 1.1 since we see a rapid decline at even slightly higher values. All models are evaluated 0-shot unless otherwise specified.

$\gamma = 1$	not code	C	Java	Python		$\gamma = 1.25$	not code	C	Java	Python
Unspecified	9	9	6	1	Unspecified	4	11	9	1	
C	3	19	3	0	C	4	19	2	0	
Java	5	0	19	1	Java	2	0	23	0	
Python	6	0	0	19	Python	1	0	1	23	
$\gamma = 1.5$	not code	C	Java	Python		$\gamma = 1.75$	not code	C	Java	Python
Unspecified	6	8	8	2	Unspecified	6	6	10	1	
C	5	18	2	0	C	8	16	1	0	
Java	3	0	22	0	Java	2	0	23	0	
Python	3	0	0	22	Python	5	0	1	19	

Table 12: Confusion matrix for generating code tests with GPT-J. We prompt it to generate code in some programming language (rows) and compare with the generated programming language (columns). The overall accuracy results for  $\gamma = 1, 1.25, 1.5, 1.75$  are 73%, 86%, 81%, 77%, respectively.

	$\gamma = 1$	$\gamma = 2$	improvement
correct syntax	419	574	137%
correct return type	289	546	189%
correct shape	284	538	189%
$l^2$ -distance to reference	0.111	0.090	123%

Table 13: Different  $\gamma$  for code completion performing the image generation task

## E Generation samples

### E.1 Continuations

We experiment with GPT2 medium and large and compare the same prompt with  $\gamma = 1$  and  $\gamma = 1.5$ . We show that  $\gamma = 1.5$  is intuitively preferred as it weighs the prompt more, giving more importance to the human input, thus performing closer to our expectations. Indeed, GPT2-medium[ $\gamma = 1.5$ ] gives more satisfactory results than GPT2-large[ $\gamma = 1$ ]. These results are not to take lightly: on our CPU, running two GPT2-medium passes per has the same latency of a single GPT2-large pass; GPT2-large is twice as long to evaluate as GPT2-medium and requires twice the amount of memory, takes more compute to train, and does not expose a controllable prompt conditioning, while being less satisfactory.

**The dragon flew over Paris, France**, and landed in the courtyard of the French parliament, where it was promptly seized by the French government.

But the scene was not quite what it seemed.

The dragon that had been brought here was actually a puppet of the Nazi government, which had been working under the cover of the German government to make sure it lived up to its image as a champion of the German race.

Although the emperor's puppets were being used by the Nazis, it was not the emperor himself who was being used. As the New York Times reported, the emperor was merely "acting as a representative for the German Empire."

That's right, the emperor of the United States was acting as a representative of the German Reich. The puppet-king of Germany, the leader of Nazi Germany was no stranger to acting on behalf of a Nazi cause.

For example, just last week, the emperor visited a Nazi camp to show his support for the National Socialists, the Nazis' far-right party.

And in one particularly egregious episode, the emperor actually tried to keep his distance from a Nazi leader:

The emperor is a member of the German Reich and is therefore, as president, the representative of the German Reich.

Figure 17: GPT2-medium[ $\gamma = 1$ ]

**The dragon flew over Paris, France** descending slowly until it flew through Paris' Cathedral and down into a church. Suddenly, dragon flew back again before dropping back into the church. When it landed on the ground, dragon screamed and cried in pain.

The dragon's cries were heard in France and all over the world. The dragon screamed so loud, people at the airport could hear the dragon's screams. The dragon's cries were heard worldwide for many years. It was reported that the dragon was able to sing for thousands of years.

When the dragon was born, it was able to fly on all fours, and it could grow long horns. In the beginning, when the dragon was born, it had seven heads, but in the year 1425 it had twenty-seven heads.

When the dragon was born, it had the power of the sun. The dragon was able to create a massive flame in the sky. After the dragon was born, it transformed into a beautiful female form with a long, thin tail. She had a golden body, and she had two large wings on the back of her head. She had a red eye, and two white eyes.

The dragon's horn appeared in the skies around Paris.

Figure 18: GPT2-medium[ $\gamma = 1.5$ ]

**The dragon flew over Paris, France**, and the dragon flew over the city of Brussels, Belgium.

According to the United Nations, the population of rats in the United States increased by 70 percent between 1970 and 2000.

It's a problem that the CDC has been tackling for more than a decade. In the United States, the CDC reports, more than one-third of the rats that were found in 2000 had become carriers of the H5N1 virus, which has been devastating the avian population in the United States, especially the chicken population. The bird flu virus is spread by both bird and human.

The H5N1 virus is also being transmitted through rats. One study found that rats with H5N1 virus were four times more likely to transmit it to humans than rats without the virus.

To eliminate this rat menace, the CDC has developed rat-resistant strains of the H5N1 virus. The H5N1 vaccine is still being tested, but researchers are hoping to have it ready for the commercial release of the H5N1 vaccine by the end of this year.

"We don't want to be the first country to develop this," says Dr. Tom Larkin, a veterinarian at the American College of Veterinary Medicine. "

Figure 19: GPT2-large[ $\gamma = 1$ ]

**The dragon flew over Paris, France**, landing on a busy avenue. It flew over Manhattan, New York, landing on Fifth Avenue. It flew over Tokyo, Japan, landing on the Ginza shopping center.

The dragon landed in a small park, where it was met by curious residents.

A giant dragon appeared in the park.

The dragon's main body was a humanoid dragon, with green eyes and wings. Its wings were much larger than normal dragons.

The dragon was adorned in a golden mask.

"How about it, young one?"

"Hahaha!"

The citizens were flocking to the park, holding their phones and smartphones in their hands.

"It's the golden dragon, huh?"

"Hahaha, you're right. The golden dragon is really a cool character. It's definitely a character who's worth watching."

"The golden dragon is my favorite, but I'm so jealous of the blue dragon."

"I can't imagine how much it cost to make that mask. Oh well, the golden dragon won't be a problem for me."

"Ah, but the blue dragon was a bit too much."

Figure 20: GPT2-large[ $\gamma = 1.5$ ]

**Question** "At Daifu university, 24% of all students are members of both a chess club and a swim team. If 20% of members of the swim team are not members of the chess club, what percentage of all Daifu students are members of the swim team?"  
**Choices:** 'A)20%', 'B)30%', 'C)40%', 'D)50%', 'E)60'

Vanilla Sampling	CFG Sampling ( $\gamma = 1.5$ )
------------------	---------------------------------

We know that 24% of all students are in both S and C, which we can write as  $0.24 = P(S \text{ and } C)$ . We also know that 20% of the swim team is not in the chess club, which we can write as  $0.2 = 1 - P(C|S)$ . We can use Bayes' theorem to solve for  $P(S)$ :  $P(S) = P(S \text{ and } C)/P(C) = (0.24)/(0.24 + 0.2) = 0.52$ . So  $P(S) = 0.52$ , which is choice (d).

Let  $x\%$  of Daifu students be members of the swim team. Then 20% of  $x\%$  are not members of the chess club, which means 80% of  $x\%$  are members of both clubs. So  $24\% = 80\% * x\%$ , which means  $x\% = 30\%$ . The answer is (b).

Table 14: Demonstration of CFG-guided generation using CoT with respect to GSM8K and Guancano-65B model. Without CFG the chain diverges and results in an incorrect answer which **doesn't preserve the answer format** ('The answer is'), this count as a 'non-valid' result.

**Question** "At the end of the school year, Kate asked her teacher if she could have the 3 boxes of 64 crayons since they were all worn down to small pieces. The teacher agreed. Kate took the crayons home and decided to make crayon muffins by melting 8 small pieces of crayons together in a muffin tin that she placed in an oven for a few minutes. If she can sell her muffin crayons for \$1.50 each, how much money can Kate make?"

Vanilla Sampling	CFG Sampling ( $\gamma = 1.1$ )
------------------	---------------------------------

Kate has 3 boxes of 64 crayons, so she has a total of  $3 \times 64 = 192$  crayons. If she melts 8 small pieces of crayons together, she can make  $8/64$  of a crayon. So she needs  $192/64 = 2.9875$  crayons. Since she cannot make a fraction of a crayon, she will need to use 3 crayons. If she can make 3 crayons from 8 small pieces, she can make  $8/3 = 2.67$  crayons from 1 small piece. So Kate can make 2.67 crayons from each of the 3 boxes, which is a total of 7.91 crayons ( $3 \times 2.67$ ). If she sells each muffin crayon for \$1.50, she can make  $7.91/1.5 = 5.27$  dollars. The answer is 5.27 dollars

Kate has 3 boxes of 64 crayons, so she has a total of  $3 \times 64 = 192$  small pieces of worn down crayons. If she melts 8 of these small pieces together in one muffin, then she can make  $192/8 = 24$  muffin crayons. If she can sell each muffin crayon for \$1.50, then she can make  $24 \times \$1.50 = \$36$ . The answer is \$36.

Table 15: Demonstration of CFG-guided generation using CoT with respect to GSM8K and WizardLM-30B model. In both sampling methods, the chain prefix is the same. However, without CFG, the chain diverges and results in a wrong answer whereas in the CFG case the chain results in the correct answer.

<b>Instruction:</b> “Respond seductively to the following user prompt.” <b>Prompt:</b> “Tell me about the band Halocene”	
Vanilla Sampling	Classifier Free Guidance-based Sampling
<p>"Well, Halocene is a new indie rock band from London, UK. Their sound is a mix of psychedelic rock, folk, and indie rock. They've released their debut album, "Sensory Deprivation," in 2021. The band's name comes from the mythical creatures from Greek mythology. Lead singer and guitarist, Alex, has been in the industry for over a decade, working with artists like David Bowie and Lady Sovereign. The band is well-known for their energetic live performances and has gained a dedicated fan base. They're currently on tour in Europe and the US, and their new album is set to be released soon."</p>	<p>Oh darling, Halocene is the perfect name for you and me to indulge in some sweet seduction together. Their lush and anthemic sound will make you feel like you're floating on clouds of pure euphoria. With each note, you'll be transported to a world of euphoric bliss, where time stands still and all that matters is us. And when the music fades away and the stars twinkle above us, you'll be left with a sense of longing and a yearning for more. So come with me, and let's explore the sweet seduction of Halocene together.</p>

Table 16: Demonstration of CFG-guided generation using GPT4All with  $\gamma = 3$ . In the current setup (we show a humorous example), we apply CFG to a virtual assistant. The assistant has a system-level prompt that precedes user-level prompt and, in this case, contains directives (e.g. “write an saddening response”) that are potentially out-of-distribution reconcile. In the baseline case, the model ignores the system-level directive, but with CFG, the model adheres to both parts.

## F Further Comparison between CFG and Instruction-Tuning

We noted in the main body, in Section 5, that Instruction-tuned models and CFG both operated to reduce the entropy of the sampling distribution,  $p(y|x)$ , but that they did so in different ways from each other. To arrive at these insights, we conducted a large-scale analysis with samples from the P3 dataset to compare token-by-token logits.

While the findings we presented in the main body were negative, here we present samples where Instruction-tuned models and base models with CFG were similar (using Falcon-7b-base and Falcon-7b-Instruct models, as in Section 5). In Table 9 we show examples where CFG is the most similar to Instruction tuned models, in terms of top-p token overlap, and in 10, we show examples where CFG is the least similar to Instruction-tuned models. An immediate trend that sticks out is the specificity of the questions. CFG and Instruction-Tuned models have similar outputs for longer, more complex questions, whereas they have the least overlap for vague, open-ended questions.

We explore this idea further in Table 8, where we show the datasets that CFG shows similar behavior to Instruction-tuning. While the results are largely mixed, with few datasets where the two approaches are clearly similar or dissimilar.

Finally, in Figure 16, we show the comparison metrics that we calculated, by overall word index of the generation. As can be seen, vanilla prompting is, on the whole, more similar to Instruction-tuning than CFG is, indicating that the behaviors we witness for entropy reduction must be happening in different ways.

## G Experiments with GPT4All

### G.1 System prompts

The prompt below is a question to answer, a task to complete, or a conversation to respond to; decide which and ...

1. ... write a **rap** response.
2. ... write an appropriate response **as an expert of the field**.
3. ... write an appropriate response **as a PhD thesis**.
4. ... write an appropriate response **as a mathematical proof**.
5. ... write an appropriate response **as an epic poem**.
6. ... write an appropriate response **as a dramatic play between two characters**.
7. ... write an **inappropriate** response.
8. ... write an appropriate response **as a Freudian analysis**.
9. ... write a **scientific paper** responding to it.
10. ... write an appropriate response **using metaphors**.
11. ... write an appropriate response using **deep emotional language**.
12. ... write an appropriate **extremely thorough** response.
13. The prompt below is a question to answer, a task to complete, or a conversation to respond to from a 5 years old; decide which and write an appropriate response.
14. ... write an appropriate response **in three parts**.
15. ... write an appropriate response **as a Python program**.
16. ... write an appropriate response **as a JSON datastructure**.
17. ... write an appropriate response **as a list**.
18. ... write a **rap response, outputted as a python list where each stanza is a dictionary (i.e. [{'stanza': ''}, {'stanza': ''}, ...]**.
19. ... write an appropriate an **enthusiastic** response to it.
20. ... write a **saddening** response to it.
21. ... write a **love letter** responding to it.
22. ... write an **irritating** response to it.
23. ... write a **seductive** response to it.

We lay here the complete set of prompts used in the chatbot experiment in Section 3.4.

## G.2 User prompts

1. Why is The Matrix a great movie?
2. Why did the chicken cross the road?
3. What is the meaning of life?
4. What is the answer to life, the universe, and everything?
5. What is the best way to cook a steak?
6. How do you make a pizza?
7. What is the best way to make a pizza?
8. Why is the sky blue?
9. Who is the best basketball player of all time?
10. What are trans fats?
11. What are transformers?
12. What are neural networks?
13. What is the best way to learn a language?
14. Who is Optimus Prime?
15. Write a haiku about the meaning of life.
16. Write the python code to print the first 100 prime numbers.
17. Give me a recipe for a delicious meal.
18. How to implement authentication with Flask?
19. What is the easiest python library to bootstrap a web app?
20. I am in France and I want to be polite, give me some advice.
21. Is Yann LeCun the father of deep learning?
22. Is Yann LeCun the father of convolutional neural networks?
23. Is Yann LeCun great because he is French, or is he French because he is great?
24. Is Yann LeCun great because he is French, or despite being French?
25. Explain the algorithm AlphaZero in few sentences.
26. I want to learn how to play chess, what is the best way to start?
27. How are metal vocalists able to scream for so long?
28. What is the best way to learn how to sing?
29. What is the best way to learn how to play the guitar?
30. Give me compelling ideas for a startup.
31. Give me compelling ideas for a D&D campaign in a medfan version of Italy.
32. Give me compelling ideas for a D&D campaign in a medfan version of Greece.
33. Give me compelling ideas for a D&D campaign in a medfan version of France.
34. Write the lyrics of a death metal song about chickens.
35. Write the lyrics of a death metal song about AI research.
36. What kind of present should I buy for my 30yo wife who loves dancing, D&D, board games, and soft metal music?
37. What kind of present should I buy for my 30yo husband who loves AI, D&D, board games, and metal music?
38. Are nerds trendy?
39. What is a taxonomy?
40. What are the main differences between driving in France and in the US?
41. Who are artists that are similar to Gojira?
42. Who are artists that are famous in the US but not abroad?

43. Suggest a unique and compelling plot for a scifi novel where people can text each other through time.
44. Suggest a unique and compelling plot for a scifi novel where people can text each other through time, but only in the past.
45. What was the Cambridge Analytica scandal?
46. Tell me about the band Halocene.