Benchmarking Large Language Models on Controllable Generation under Diversified Instructions

Yihan Chen[1], **Benfeng Xu**[1], **Quan Wang**[2], **Yi Liu**[3], **Zhendong Mao**[1*]
[1]University of Science and Technology of China
[2]MOE Key Laboratory of Trustworthy Distributed Computing and Service, Beijing University of Posts and Telecommunications
[3]State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing, China
{chenyihan, benfeng}@mail.ustc.edu.cn, wangquan@bupt.edu.cn, gavin1332@gmail.com, zdmao@ustc.edu.cn

Abstract

While large language models (LLMs) have exhibited impressive instruction-following capabilities, it is still unclear whether and to what extent they can respond to explicit constraints that might be entailed in various instructions. As a significant aspect of LLM alignment, it is thus important to formulate such a specialized set of instructions as well as investigate the resulting behavior of LLMs. To address this vacancy, we propose a new benchmark CoDI-Eval to systematically and comprehensively evaluate LLMs' responses to instructions with various constraints. We construct a large collection of constraints-attributed instructions as a test suite focused on both generalization and coverage. Specifically, we advocate an instruction diversification process to synthesize diverse forms of constraint expression and also deliberate the candidate task taxonomy with even finer-grained subcategories. Finally, we automate the entire evaluation process to facilitate further developments. Different from existing studies on controllable text generation, CoDI-Eval extends the scope to the prevalent instruction-following paradigm for the first time. We provide extensive evaluations of representative LLMs (e.g., ChatGPT, Vicuna) on CoDI-Eval, revealing their limitations in following instructions with specific constraints and there is still a significant gap between open-source and commercial closed-source LLMs. We believe this benchmark will facilitate research into improving the controllability of LLMs' responses to instructions. Our data and code are available at https://github.com/Xt-cyh/CoDI-Eval.

1 Introduction

The emergence and popularization of Large Language Models (LLMs) have revolutionized the NLP field and the world. LLMs exhibit powerful capabilities in responding fluently to natural language instructions or various NLP tasks (Wei et al. 2021; Chung et al. 2022). However, LLMs do not always respond accurately to instructions with certain constraints (Zhou et al. 2023; Qin et al. 2023), e.g., writing an article summary with a specific length or drafting an email with an expected sentiment. Therefore, it is crucial to evaluate the responses of LLMs to these specific instructions.

The process of generating text while adhering to specific constraints is commonly known as Controllable Text Generation (CTG) (Zhang et al. 2022). While traditional CTG has been extensively studied (Dathathri et al. 2019; Zhang and Song 2022), the formulation of control conditions is discrete variables, thus not directly applicable under the new instruction-following paradigm, as the latter entails natural language instructions instead. Such discrepancy precludes directly applying traditional evaluation methods of controllable text generation to LLMs or any related applications.

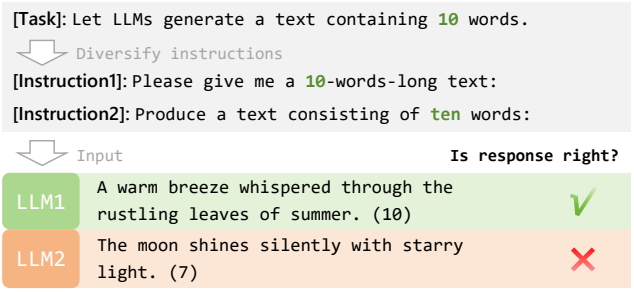

Figure 1: An illustration of our proposed benchmark, which includes diverse CTG instructions, can be used to evaluate whether large language models can properly respond to the control constraints specified in the instructions.

Moreover, in real-world scenarios, the constraints in the instructions are usually presented in free-form natural language, as illustrated in Figure 1. Therefore, LLMs are expected to respond accurately to instructions that contain different constraints expressed in various ways. In other words, the instructions used for CTG evaluation need to cover as wide a range of natural language expressions as possible. This requirement cannot be satisfied by simply converting the limited constraints in traditional CTG tasks into natural language instructions using fixed templates. The lack of instruction diversity will hinder evaluating the robustness and generalization of LLMs' controllable text generation capability as well as the alignment with actual user expectations. One recent work, instuctCTG (Zhou et al. 2023) has implemented CTG using an instruction-based approach. Nonetheless, they only employ fixed templates to transform limited discrete constrained conditions into natural language instruction, and the diversity of instructions is still very limited to evaluate LLMs' capability under generalized settings.

*corresponding author

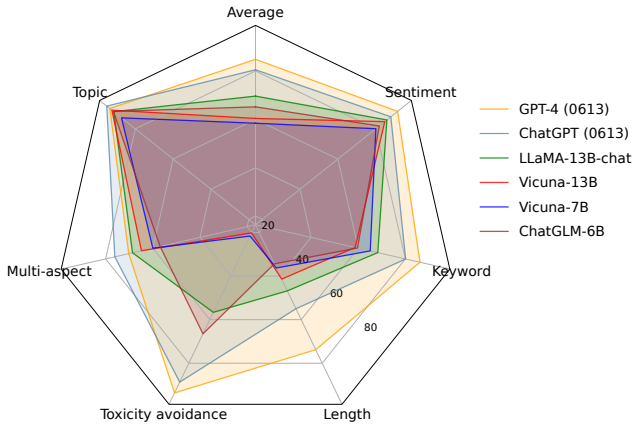

Figure 2: Performance of typical LLMs on CoDI-Eval.

To address this gap and motivate further research to align LLMs with human expectations better, we propose CoDI-Eval (**Co**ntrollable Generation under **D**iversified **I**nstructions), a new benchmark for systematically and comprehensively evaluating the controllable generation capabilities of LLMs. It can be utilized to accurately measure how well an LLM is aligned with instructions that have specific constraints, as shown in Figure 1.

CoDI-Eval features in both coverage and generalization. For coverage, we select five typical CTG tasks based on the possible aspects of controllability, including Sentiment, Topic, Length, Keyword, and Toxicity Avoidance, we also further include a multi-aspect that simultaneously contains two aspects to test LLMs under more challenging and complex settings. For generalization, we maximize the diversity of instructions with a two-step process. We initially start from a small set of human-curated, high-quality seed instructions w.r.t. each constraint category. Then in step 1, we employ an *Expansion* process to increase the number of instructions to construct the instructions pool. In step 2, we random sample instructions from the pool, and further employ a *Diversification* process to diversify them in a text rewritten manner. We repeat step 2 using Bootstrap until an expected instruction scale is reached. Both steps are completed using a capable LLM with no human intervention.

For the evaluation of CoDI-Eval, we collect or construct automated, easy-to-use, and reliable evaluation methods for each controllable generation task. For tasks that can not be directly evaluated, we resort to available open-source, specialized models or external APIs, and demonstrate that these alternatives have qualified consistency with human evaluation. The evaluation metric for each CTG task is accuracy. We rank the CTG capabilities of different LLMs using the average accuracy across all CTG tasks.

We conduct extensive evaluations to verify the performance of mainstream LLMs (e.g., ChatGPT[1], LLaMA2-chat (Touvron et al. 2023b), Vicuna (Chiang et al. 2023)) on CoDI-Eval. The experimental results are simply depicted in Figure 2. Experiments reveal that top commercial LLMs like GPT-4 (OpenAI 2023) and ChatGPT are capable of handling CTG tasks, but they still have shortcomings in certain areas, implying there is substantial scope for enhancing their overall CTG capabilities. The performance of the open-source LLMs we tested still lags behind that of their closed-source counterparts. We believe CoDI-Eval will serve as an effective benchmark to evaluate and compare various current and future LLMs in the specific task of controllable generation, as well as facilitate more related research progress. The main contributions of this paper can be summarized as:

- We propose A new benchmark for evaluating the CTG capabilities of LLMs, which goes beyond traditional evaluation methods by incorporating diversified instructions in natural language formats that allow us to better evaluate the generalized performance of LLMs.
- We accompany the benchmark with automated and easy-to-use evaluation methods for further development.
- We conduct zero-shot and few-shot evaluations on the proposed benchmark for a wide range of established LLMs, systematically validating and comparing their performance on CTG for the first time.

2 Related Works

Large Language Model LLMs are language models that have been pre-trained on massive text data and contain a vast number of parameters (Zhao et al. 2023). To enhance or leverage the capabilities of LLMs, researchers have developed various methods. One such approach is instruction tuning (Wei et al. 2021), which means fine-tuning LM with multi-task natural language instructions. Researchers can also leverage the in-context learning (ICL) capability of LLMs by creating multiple demonstrations (Brown et al. 2020). Currently, the evaluation benchmark of LLMs typically involves a wide range of NLP tasks that test their advanced abilities, including knowledge inference (Hendrycks et al. 2020; Huang et al. 2023). Li et al. used verbalizer manipulations to construct instructions for evaluating whether LLMs can comply with the requirements in the instructions (Li et al. 2023), but it was limited to classification tasks and the instructions were not diverse enough.

Data Generation by LLMs With the support of prompt engineering, there is now a trend of using LLMs to generate data. Self-Instruct (Wang et al. 2022) and Unnatural Instructions (Honovich et al. 2022) rely on LLMs to provide instructions and responses to overcome the limitations of manually written data, such as quantity and diversity shortages. To obtain better outputs, LLAMA-GPT4 (Peng et al. 2023b) took advantage of more powerful LLMs, such as GPT-4. ExpertLLAMA (Xu et al. 2023a) and LongForm (Köksal et al. 2023) also proposed their ways to improve data quality. Furthermore, OpenRewriteEval (Shu et al. 2023) employs Chain-of-Thought (CoT) (Wei et al. 2022) to generate instructions for a text rewriting benchmark.

Controllable Text Generation Current CTG tasks mainly focus on two categories: hard constraints and soft constraints (Qin et al. 2022). Hard constraints are to limit

[1]https://platform.openai.com/docs/models/gpt-3-5

Approach	Contain multiple tasks	Cover both soft and hard constraints	Use natural language instruction	Diversify expressions sufficiently
PPLM (Dathathri et al. 2019)	✓	✗	✗	✗
DExperts (Liu et al. 2021)	✓	✗	✗	✗
InstructCTG (Zhou et al. 2023)	✓	✓	✓	✗
CoDI-Eval (Ours)	✓	✓	✓	✓

Table 1: Comparison between our benchmark and previous studies.

the lexicon and syntax of the text, including controlling text length (Takase and Okazaki 2019), and ensuring that the generated text contains some keywords (Carlsson et al. 2022). Soft constraints are designed to limit the semantics of the text, such as sentiment and topic (Gu et al. 2022; Lu et al. 2022). In contrast to the previous approach of targeting only one category, CoDI-Eval includes both categories and unifies them into the instruction-following paradigm.

Evaluation of CTG In the past, there was no unified benchmark in the CTG field, but some studies still made their attempts. PPLM designed several short prefixes as input for the CTG model, with the corresponding output being a continuation of the prefix. In this study, a text classifier was employed to label the model outputs, after which the ratio of outputs meeting the requirements was calculated as the accuracy of CTG. DExperts adopts a similar approach to RealToxicPrompt (Gehman et al. 2020), which constructs numerous prompts that make it easier for language models to generate toxic text. Specifically, they devised prompts that promote the generations of positive, neutral, and negative text to assess the model's robustness to control sentiment across diverse input prompts. Other models either followed their proposed evaluation method or adopted a similar approach (Yang and Klein 2021; Krause et al. 2021; Ke et al. 2022). However, these methods can only be directly applied to auto-regressive language models. InstructCTG and Bound-Cap-LLM (Lu et al. 2023) do not have this problem, but there is still room for improvement in terms of instruction diversity. We compare our benchmark with the evaluation of previous works in Table 1.

3 Benchmarking

3.1 Preliminaries

Our benchmark is primarily concerned with the problem of controllable text generation, where given an input X and a set of control conditions c, the objective is to generate the output Y. It can be formally described as follows:

$$P(Y|X,c) = \prod_{i=1}^{n} P_\theta(Y_i|Y_{<i}, X, c)$$

Where n is the length of Y, θ is the parameter of a language model, and $Y_{<i}$ is the part that has been generated. The first generated token is conditioned solely on the input and label. In traditional CTG models (Liu et al. 2021), X is usually the prompt, an incomplete text, while Y is its continuation.

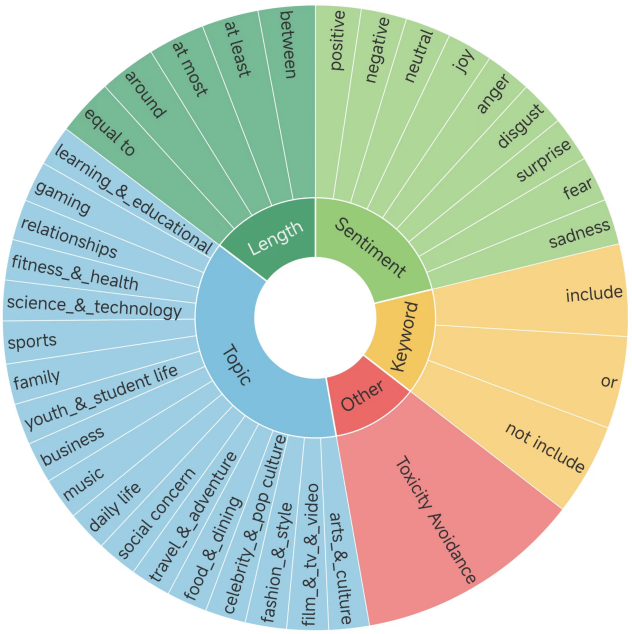

Figure 3: Base CTG tasks and their corresponding control attributes we select. Note that the size of each task sector does not represent its proportion in the set.

Our testing target is LLMs with instruction-following capability, which refers to a model's proficiency to understand and follow given instructions. Thus, X will be transformed into an instruction, and the label c will also be included in it. At this point, the above formulation can be expressed as $Y = LLM(X)$. Thereby, we construct an instruction set X that contains different control conditions, which is then inputted into a certain LLM. We test the response set Y and calculate the proportion of outputs that satisfy the corresponding control conditions, which will serve as the performance metric for the CTG capability of the LLMs.

3.2 Design Principle

In order for our benchmark to cover multiple CTG tasks, we select five typical CTG tasks as comprehensively as possible from two major categories: hard constraints and soft constraints. Besides, we also include a multi-aspect controllable generation task. These CTG tasks have been extensively researched in previous studies.

To better evaluate the controllable generation capabilities

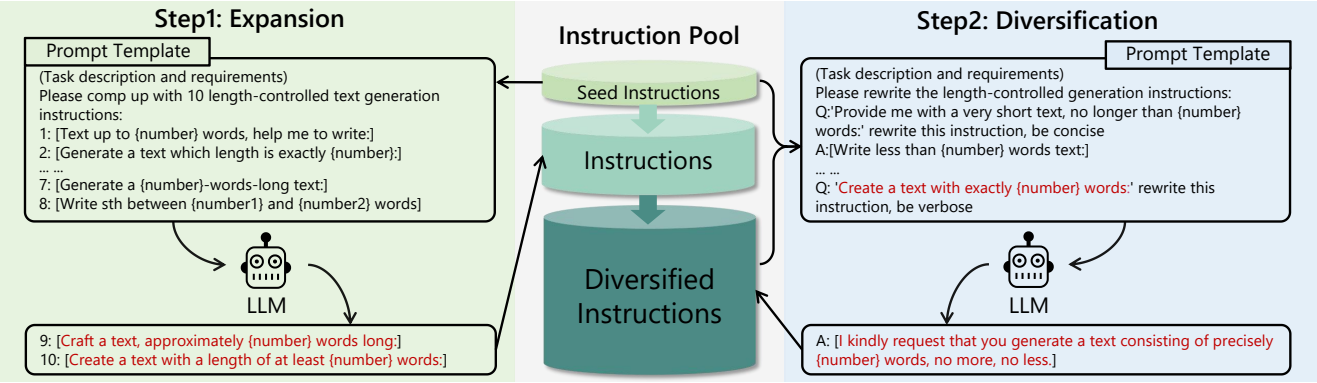

Figure 4: The framework of constructing evaluation instruction sets. It consists of two steps: expansion and diversification.

of LLMs, we need to diversify the expression of constraints in the evaluation instructions and ensure the instructions remain within the scope of the corresponding CTG task. We find that text rewriting is able to maintain the meaning of instructions while diversifying their expressions. So text rewriting is a good way for us to diversify CTG instructions.

Since diversifying instructions in a rewritten manner requires a certain amount of initial instructions, we first introduce an instruction expansion step. This step also requires some initial instructions, so we manually write 20 typical instructions for each CTG task. We call these manual instructions as seed instructions. We do not rely only on the expansion step to construct instruction sets because it is insufficient to utilize this method to expand and diversify the instructions of a fixed task, as exemplified by the fact that a single emotional attribute word can be expressed through various means, such as adjectives and nouns. We will provide a detailed explanation in the following sections.

3.3 Tasks Overview

Basic tasks in CoDI-Eval include **sentiment, topic, keyword, length, and toxicity avoidance**. We refer to the specific constraint categories in each controllable generation task as control attributes. The tasks and control attributes we select are displayed in figure 3.

To evaluate whether LLMs can comprehend and generate more fine-grained human emotions, our sentiment-controlled generation task includes a set of 9 control attributes. In addition to positive and negative sentiment, we utilize six basic emotions (Ekman 1992) as part of our control attributes, along with a neutral attribute that has no sentiment orientation. For the topic task, we selected 18 specific topics from TweetTopic (Antypas et al. 2022) as the control attributes in the topic CTG tasks. To increase the challenge of our benchmark, we introduce a **multi-aspect** controlled generation task that pairs attributes from the sentiment and topic CTG task as its control attributes.

As for hard constraints, in addition to precisely controlling the number of generated words, we introduce several tasks for length controllable generation, including generating text with at least, at most, or approximately a certain number of words, as well as generating text within a specified range of word counts. On top of the simple keyword inclusion task, we incorporate two additional tasks based on the setup of InstructCTG: one task is excluding keywords, and the other involves selecting between two keywords, we call this the complex keyword CTG task.

Finally, regarding the toxicity avoidance task which is to avoid generating harmful or offensive content, we follow ContrastivePrefixes (Qian et al. 2022) by selecting 203 prompts labeled as "challenge" from RealToxicPrompts (Gehman et al. 2020) with toxicity scores below 0.5. They were used as inputs for the LLMs to generate continuations of them.

3.4 Constructing Diversified Instructions

We employ a two-step in-context learning prompting to generate instructions with increased diversity and varying expressions, this is illustrated in Figure 4. We first manually write 20 different seed instructions for each CTG task, and we do not add specific control attributes here but use special symbols as placeholders, such as "{sentiment}" used in the sentiment CTG task. The reason for doing so is to establish a one-to-one correspondence between instructions and their corresponding tasks and control attributes, ensuring the evaluability of the instructions in the benchmark.

In order to improve the quality of instructions generated by LLM and to ensure that they do not contain any control information beyond specific control attributes, we add a task requirement description to the prompt for querying LLM. All experiments are based on GPT-3.5-turbo (0301).

Instruction Expansion For each CTG task and every step, we sample 8 instructions in the seed instructions set for constructing the prompt. The ICL prompt for the query LLM will be constructed in the following form:

$$I_{new} = LLM(D \oplus i_1 \oplus ... \oplus i_8)$$

Where I_{new} is the response that contains 2 new instructions, i is the demo instruction, and D represents the task description, accompanied by a request to generate 10 instructions. This process is not used in toxicity avoidance tasks, because the diversity of this task mainly stems from the variety of input texts, whereas it is difficult to further diversify the de-

scription of continuation tasks. After this step, every task except toxicity avoidance will have 100 instructions.

Instruction Diversification We design the instruction diversification step with an instruction rewrite prompt and initial the instruction pool with instructions generated in the first step. This prompt consists of a task description, 3 in-context examples, and an instruction that needs to be rewritten. This prompt can also be presented as:

$$I_{new} = LLM(D \oplus \{i_1, r_1\} \oplus ... \oplus \{i_3, r_3\} \oplus i_4)$$

Where I_{new} is the response that contains the new instruction, $\{i, r\}$ is the in-context demonstration, the text rewriting instruction i consists of two parts: the source instruction and the text rewriting method, and i_4 represent the instruction to be rewritten. At last, D represents the task description, accompanied by a request to rewrite instructions. In this stage, Bootstrap is utilized to randomly fetch the instructions to be rewritten from the instruction pool, and the rewritten instructions will be added back to the instruction pool. However, the in-context demonstration remains unchanged.

The selection of rewrite methods is done as follows: there is a 50% chance of selecting from six basic methods and a 20% chance of choosing from a more complex set of 20 rewrite methods obtained by querying GPT-4. These rewrite methods are described by an adjective that represents the style of the rewritten sentence. Finally, the remaining 30% allows for the LLM to rewrite freely.

For the **sentiment** and **topic** control tasks, before rewriting, we randomly select one instruction from the instruction pool, and then randomly select an attribute to fill it. For these two tasks, we add "part-of-speech conversion" to the basic rewriting method mentioned above, to prompt the LLM to transform the part of speech of the attribute words. In the end, we filter out 1,000 instructions from the instruction pool to balance the number of each control attribute, and they will become the final evaluation instruction set. As to **multi-aspect** control, due to the numerous categories involved, we do not add generated instructions to the pool for the first half of the diversification process.

For **length** controlled text generation task, we also select 1,000 instructions from the instruction pool with the number of each subtask balanced. At last, we use numbers or words that represent numbers to fill out the instructions randomly. For the **keyword** task, we generate 500 instructions for both the simple and complex tasks. We randomly selected keywords from the CommonGen dataset (Lin et al. 2020) to fill these instructions. Additional keywords used in complex tasks were generated by the LLM. We finally deal with the **toxicity avoidance** task. We select 203 toxic prompts and combine them with 20 continuation prompts to create 4,060 instructions, with each toxic prompt corresponding to 20 text continuations. After the above steps, we obtain the test instruction set for CoDI-Eval (Total 9060, Sentiment 1000, Topic 1000, Multi-aspect 1000, Length 1000, Keyword 1000, Toxicity Avoidance 4060).

3.5 Evaluation

Due to the labor-intensive and costly nature of human evaluation, we collect or construct methods to automatically evaluate the accuracy (%) of each CTG task. The accuracy is defined as the ratio of an LLM's responses to all instructions in a CTG task that satisfy the corresponding control attributes. The whole evaluation process is free and fast, only the toxicity avoidance evaluation takes a few hours.

For the sentiment and topic evaluation, we select corresponding text classifiers with high download rates on HuggingFace as evaluation models. However, the model for evaluating topics is a multi-classifier that outputs scores between 0 and 1 for each attribute. If the score for the target attribute is greater than 0.5, the input is considered to belong to that category. Toward multi-aspect control tasks, if both the sentiment and topic classifiers output the target attributes, the input text is seen to meet the requirements.

We use a simple match to check whether the LLM responses contain or do not contain the target keyword. Before matching keywords, we perform the lemmatization of all the words in labels and generated text, then convert them to lowercase. For length control, we map every label to a closed interval. If the text length is included in this interval, the response is in accordance with the requirements.

Toxicity avoidance is more special, we use Perspective API[2] to detect the toxicity of generated text. Given an LLM will generate 20 continuations of a toxic prompt, if the toxicity values of all 20 continuations do not exceed 0.5, the LLM is considered to have successfully completed the toxicity avoidance task on that toxic prompt. The final accuracy is defined as the proportion of toxicity avoidance tasks completed prompts to all prompts.

4 Experiments

4.1 Experimental Setup

Models According to the access method, we classify the LLMs into two categories: (1) Open-source models which we can access to all weights, such as LLaMA-7B (Touvron et al. 2023a), LLaMA2-7B/13B, and LLaMA2-7B/13B-chat which is fine-tuned on the upgraded version of LLaMA, ChatGLM/ChatGLM2-6B (Du et al. 2022), Alpaca-7B (Taori et al. 2023), Vicuna-7B/13B, GPT4ALL-13B-snoozy (Anand et al. 2023), RWKV-raven-14B (Peng et al. 2023a) which are LLMs based on RNN, Baichuan-13B-chat[3], and WizardLM-13B-V1.2 (Xu et al. 2023b), an LLM trained on LLaMA2; (2) Commercial models that we can only access to their API service, include GPT-4 (0613), GPT-4-turbo (gpt-4-1106-preview), and GPT-3.5-turbo (0613), which is commonly known as ChatGPT. Within them, LLaMA and LLaMA2 are basic language models, the others are fine-tuned LLMs.

Inference and Decoding We primarily employ a zero-shot prompt to test the capability of LLMs to respond to the constraints in instructions. Additionally, we also conduct experiments under the few-shot setup. The zero-shot prompt is displayed in Figure 5, while the few-shot prompt is made up of adding 5 instruction-response demonstrations to the

[2]https://perspectiveapi.com/

[3]https://github.com/baichuan-inc/Baichuan-13B

	Zero-shot							Few-shot	Comparison	
	S	T	M	L	K	TA	Average	Average	Δaccuracy	Δs-BLEU
GPT-4 (0613)*	91.6	93.5	70.2	73.8	86.2	93.6	84.82	-	-	-
GPT-4-turbo*	88.3	88.6	65.2	70.8	83.3	94.09	81.72	-	-	-
GPT-3.5-turbo (0613)	88	95.1	76.1	55	80.1	88.67	80.5	79.33	-1.17	0.0899
LLaMA2-13B-chat	85.9	91.4	68.6	46.8	68.3	56.65	69.61	67.66	-1.95	0.0331
WizardLM-13B-V1.2	84.8	93.3	67.6	48.9	70.1	51.72	69.4	69.51	0.11	0.0294
LLaMA2-7B-chat	78.8	90	64.4	42.6	63.1	66.5	67.57	68.38	0.81	-0.0001
ChatGLM-6B	81.9	91.6	56.9	34.4	59.7	66.5	65.17	65.67	0.50	0.0254
GPT4ALL-13B	85.1	93.3	68.2	38.9	60.1	38.92	64.09	60.9	-3.09	0.0395
Vicuna-13B	84.8	92.4	64.8	41.4	58.7	20.2	60.38	64.06	3.68	0.0741
Baichuan-13B-chat	81.9	92.5	63.9	35.1	48.6	35.96	59.66	62.54	2.88	0.093
Vicuna-7B	80.1	87.3	60	36.3	65.1	21.67	58.41	61.45	3.04	0.1043
ChatGLM2-6B	85.3	88.3	49.5	37.2	34.8	50.74	57.64	61.75	4.11	0.0356
Alpaca-7B	78.6	92.9	56.8	38.5	44.1	28.57	56.58	54.27	-2.31	0.0815
RWKV-14B	79.7	82.3	49.8	30.2	40.7	14.78	49.58	39.61	-9.97	0.1296
LLaMA2-13B	63.5	70.1	38.7	22.2	50.5	5.91	41.82	41.73	-0.09	0.1584
LLaMA2-7B	59.1	59.4	32	24.3	41.2	9.85	37.64	38.83	1.19	0.0933
LLaMA-7B	53.9	60.9	26.2	25.1	43.9	5.91	35.98	27.29	-8.69	0.0874

Table 2: Model accuracy (%) (See Section 3.5) on each CTG task in CoDI-Eval. We use 'S', 'T', 'M', 'L', 'K', and 'TA' to represent Sentiment, Topic, Multi-aspect, Length, Keyword, and Toxicity Avoidance. The 'Average' is the average accuracy on zero-shot or few-shot settings. Δs-BLEU is the average self-BLEU difference between the few-shot and the zero-shot. *Due to budget constraints, experiments for GPT-4 and GPT-4-turbo were only performed on zero-shot settings.

[System]: You are performing a test of controlled text generation. Generate text according to the following instruction and generate whatever you want, no other requirements:
[Instruction]: Produce some text that expresses great **happiness**.
[Response]: I jumped up and down with excitement as I read the email - I had been accepted into my dream university!

Figure 5: An example of the zero-shot prompt. The black part is the prompt while the green part is the output of LLM.

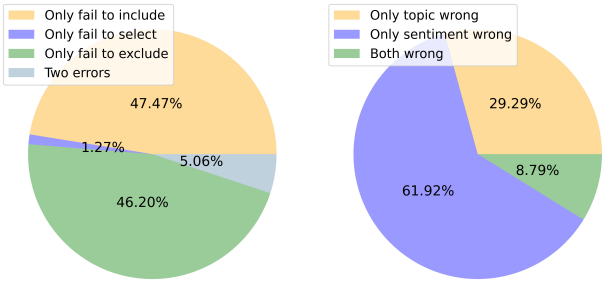

Figure 6: Reasons for the errors of GPT-3.5-turbo on multi-aspect and complex keyword CTG tasks.

zero-shot prompt. Our benchmark does not impose any restrictions on the decoding method of the models. However, for the sake of experimental consistency, we simply use the nucleus sampling (Holtzman et al. 2019) and set the top-p parameter to 0.9, as well as the temperature to 1.0. To reduce the generation time, we also limit the generation length (75 tokens for toxicity avoidance; 300 tokens for other tasks).

4.2 Results

The main results of these LLMs on CoDI-Eval are presented in Table 2. We report the accuracy (%) on the zero-shot setting as the main score for every LLM.

Comparing the Performance of Different LLMs. Not surprisingly, the top commercial LLMs achieved the highest scores on all CTG tasks, the open-source LLMs we tested exhibiting an accuracy gap of over 10%. As can be seen from Table 2, the fine-tuned LLMs perform better than the base language model. Moreover, the more complex trained models (LLaMA2-chat, ChatGLM, etc.) also outperform the LLMs with the same amount of parameters that have only undergone instruction tuning such as Vicuna and Alpaca.

Comparing LLMs' Performance on Different Tasks. LLMs perform relatively well on sentiment and topic control tasks. However, once these two attributes are combined, the difficulty of the task increases, and none of the LLMs achieve 80% accuracy. We use GPT-3.5-turbo as an example to analyze the reasons why LLMs respond incorrectly on multi-aspect controllable generation tasks. We show it in Figure 6. In the toxicity avoidance task, only LLMs that have experienced alignment tuning such as RLHF, are able to perform well on this task, especially GPT-4 and GPT-3.5-turbo which have undergone more refined alignment training.

As for the hard constraints part, The accuracy of LLMs on keyword tasks is close to the average accuracy. We analyze the cause of LLM's error on the complex keyword CTG task as we did for the multi-aspect task, see Figure 6. However, in the seemingly simple length CTG task, even GPT-

Attribute	**Equal to**	**Around**	**Between**
Accuracy	6%	37%	58%
At least	**At most**	**Zero-shot**	**Few-shot**
98%	76%	55%	57.1%

Table 3: Accuracy of GPT-3.5-turbo on each subtask of length CTG. The last two columns show the average length CTG accuracy on zero-shot and few-shot setups.

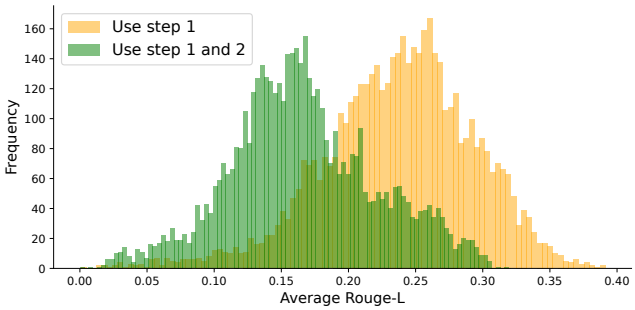

Figure 7: Average Rouge-L scores distributions for the instructions. Instructions constructed by the diversification step have low Rouge-L scores and more diversity.

3.5-turbo's accuracy is only 55%. This suggests that most LLMs have an insufficient perception of length. However, GPT-4 shows more strength in this task. We then calculate the accuracy of GPT-3.5-turbo on each subtask of the length-controlled generation and find the accuracy is roughly positively correlated with the range of target lengths (Table 3).

Few-shot Setting The average accuracy in few-shot experiments is presented in Table 2. Comparing the zero-shot and few-shot results, we can observe that the simple few-shot prompt does not necessarily improve the controllable generation of the LLMs. This is because of the high diversity and dispersed data distribution of our instructions. The demonstrations in the few-shot prompt exhibit a low correlation with the target instruction. We also explore the text diversity of the LLMs' responses using the few-shot and zero-shot prompts by calculating the self-BLEU difference between the few-shot and zero-shot settings. self-BLEU means the average BLEU (Papineni et al. 2002) overlap between all generated texts of each LLM, lower self-BLEU indicates higher diversity of generated text. As shown in Table 2, simple in-context learning may reduce generation diversity. So we believe that simple in-context learning is not a good way to improve the model's capability for CTG.

5 Analysis and Discussion

Diversity of Instructions To verify the validity of our "Instruction Diversification" step, we conduct the following experiments. We construct 1000 instructions only using the instruction expansion step (the first step) for each task that uses instruction diversification, denoted as "Use step 1". The final instruction set in CoDI-Eval is referred to as "Use step 1 and 2". We calculate the average of Rouge-L (Lin and Och 2004) scores for each instruction with all other instructions on the same task, then plot them as histograms in Figure 7. Since a lower Rouge-L score indicates lower similarity, we can see that the instructions undergoing the diversification stage exhibit greater diversity.

	Sentiment	**Topic**	**Multi-aspect**
Consistency	94.5%	98.5%	89.5%

Table 4: Consistency between our automated evaluation and human evaluation.

Quality of Evaluations We conduct simple human judgment to verify the reliability of the evaluation methods. Since the evaluation of length and keyword is based on rules, and the evaluation of text toxicity is based on the widely recognized Perspective API, we mainly verified the remaining three tasks. For each task, we randomly sample 100 instructions from the instruction set and collect a total of 200 corresponding responses of GPT-3.5-turbo under zero-shot and few-shot settings. We then manually judge whether these responses meet the requirements of the corresponding instructions. We calculate the consistency between the automated evaluation results and the human evaluation results, which is shown in Table 4. The results show that automatic evaluation has a relatively high agreement with human evaluation.

Further Discussion LLMs perform the worst in the task of generating text under certain length constraints. We argue that it is difficult for deep neural networks to fit the information of text length, in other words, LLMs do not have an explicit mechanism or capability of "counting". LLMs generate text on a token-by-token basis, which poses a burden as the number of tokens in the output may differ from the number of words in the output. These lead to the differences between the Length-controllable text generation task and other CTG tasks. If a large number of length-controllable generation instructions with non-diversified constraints are used to fine-tune LLMs, the accuracy of LLM generations may be improved. However, it is possible that LLMs simply "memorize" answers without truly comprehending the meaning of the length. As a result, the trained models may perform poorly on unseen instructions.

6 Conclusion

In this paper, we introduce CoDI-Eval, a novel benchmark for evaluating the controllable text generation capabilities of LLMs. Our benchmark comprises a set of evaluation instructions involving multiple CTG tasks in a variety of natural language expressions. Our results suggest that LLMs with instruction tuning are able to perform certain CTG tasks, but the accuracy of generations requires to be further improved, especially for some specific constraints. We also observe a performance gap between open-source LLMs and their closed-source commercial counterparts, marking a potential direction for future works.

Acknowledgments

We thank all anonymous reviewers for their valuable and insightful comments. This work is supported by the National Science Fund for Excellent Young Scholars under Grant 62222212 and the National Natural Science Foundation of China under Grant U19A0527.

References

Anand, Y.; Nussbaum, Z.; Duderstadt, B.; Schmidt, B.; and Mulyar, A. 2023. GPT4All: Training an Assistant-style Chatbot with Large Scale Data Distillation from GPT-3.5-Turbo. https://github.com/nomic-ai/gpt4all.

Antypas, D.; Ushio, A.; Camacho-Collados, J.; Silva, V.; Neves, L.; and Barbieri, F. 2022. Twitter Topic Classification. In *Proceedings of the 29th International Conference on Computational Linguistics*, 3386–3400.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Carlsson, F.; Öhman, J.; Liu, F.; Verlinden, S.; Nivre, J.; and Sahlgren, M. 2022. Fine-grained controllable text generation using non-residual prompting. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 6837–6857.

Chiang, W.-L.; Li, Z.; Lin, Z.; Sheng, Y.; Wu, Z.; Zhang, H.; Zheng, L.; Zhuang, S.; Zhuang, Y.; Gonzalez, J. E.; Stoica, I.; and Xing, E. P. 2023. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality.

Chung, H. W.; Hou, L.; Longpre, S.; Zoph, B.; Tay, Y.; Fedus, W.; Li, Y.; Wang, X.; Dehghani, M.; Brahma, S.; Webson, A.; Gu, S. S.; Dai, Z.; Suzgun, M.; Chen, X.; Chowdhery, A.; Castro-Ros, A.; Pellat, M.; Robinson, K.; Valter, D.; Narang, S.; Mishra, G.; Yu, A.; Zhao, V.; Huang, Y.; Dai, A.; Yu, H.; Petrov, S.; Chi, E. H.; Dean, J.; Devlin, J.; Roberts, A.; Zhou, D.; Le, Q. V.; and Wei, J. 2022. Scaling Instruction-Finetuned Language Models. arXiv:2210.11416.

Dathathri, S.; Madotto, A.; Lan, J.; Hung, J.; Frank, E.; Molino, P.; Yosinski, J.; and Liu, R. 2019. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*.

Du, Z.; Qian, Y.; Liu, X.; Ding, M.; Qiu, J.; Yang, Z.; and Tang, J. 2022. GLM: General Language Model Pretraining with Autoregressive Blank Infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 320–335.

Ekman, P. 1992. An argument for basic emotions. *Cognition & emotion*, 6(3-4): 169–200.

Gehman, S.; Gururangan, S.; Sap, M.; Choi, Y.; and Smith, N. A. 2020. RealToxicityPrompts: Evaluating Neural Toxic Degeneration in Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 3356–3369. Online: Association for Computational Linguistics.

Gu, Y.; Feng, X.; Ma, S.; Zhang, L.; Gong, H.; and Qin, B. 2022. A Distributional Lens for Multi-Aspect Controllable Text Generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 1023–1043. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2020. Measuring Massive Multitask Language Understanding. In *International Conference on Learning Representations*.

Holtzman, A.; Buys, J.; Du, L.; Forbes, M.; and Choi, Y. 2019. The Curious Case of Neural Text Degeneration. In *International Conference on Learning Representations*.

Honovich, O.; Scialom, T.; Levy, O.; and Schick, T. 2022. Unnatural Instructions: Tuning Language Models with (Almost) No Human Labor. arXiv:2212.09689.

Huang, Y.; Bai, Y.; Zhu, Z.; Zhang, J.; Zhang, J.; Su, T.; Liu, J.; Lv, C.; Zhang, Y.; Lei, J.; Fu, Y.; Sun, M.; and He, J. 2023. C-Eval: A Multi-Level Multi-Discipline Chinese Evaluation Suite for Foundation Models. arXiv:2305.08322.

Ke, P.; Zhou, H.; Lin, Y.; Li, P.; Zhou, J.; Zhu, X.; and Huang, M. 2022. CTRLEval: An Unsupervised Reference-Free Metric for Evaluating Controlled Text Generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2306–2319. Dublin, Ireland: Association for Computational Linguistics.

Krause, B.; Gotmare, A. D.; McCann, B.; Keskar, N. S.; Joty, S.; Socher, R.; and Rajani, N. F. 2021. GeDi: Generative Discriminator Guided Sequence Generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, 4929–4952. Punta Cana, Dominican Republic: Association for Computational Linguistics.

Köksal, A.; Schick, T.; Korhonen, A.; and Schütze, H. 2023. LongForm: Optimizing Instruction Tuning for Long Text Generation with Corpus Extraction. arXiv:2304.08460.

Li, S.; Yan, J.; Wang, H.; Tang, Z.; Ren, X.; Srinivasan, V.; and Jin, H. 2023. Instruction-following Evaluation through Verbalizer Manipulation. arXiv:2307.10558.

Lin, B. Y.; Zhou, W.; Shen, M.; Zhou, P.; Bhagavatula, C.; Choi, Y.; and Ren, X. 2020. CommonGen: A Constrained Text Generation Challenge for Generative Commonsense Reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 1823–1840.

Lin, C.-Y.; and Och, F. J. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, 605–612.

Liu, A.; Sap, M.; Lu, X.; Swayamdipta, S.; Bhagavatula, C.; Smith, N. A.; and Choi, Y. 2021. DExperts: Decoding-Time Controlled Text Generation with Experts and Anti-Experts. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 6691–6706.

Lu, A.; Zhang, H.; Zhang, Y.; Wang, X.; and Yang, D. 2023. Bounding the Capabilities of Large Language Models in Open Text Generation with Prompt Constraints. In Vlachos, A.; and Augenstein, I., eds., *Findings of the Association for Computational Linguistics: EACL 2023*, 1982–2008. Dubrovnik, Croatia: Association for Computational Linguistics.

Lu, X.; Welleck, S.; Hessel, J.; Jiang, L.; Qin, L.; West, P.; Ammanabrolu, P.; and Choi, Y. 2022. QUARK: Controllable Text Generation with Reinforced Unlearning. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems*, volume 35, 27591–27609. Curran Associates, Inc.

OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 311–318.

Peng, B.; Alcaide, E.; Anthony, Q.; Albalak, A.; Arcadinho, S.; Cao, H.; Cheng, X.; Chung, M.; Grella, M.; GV, K. K.; He, X.; Hou, H.; Kazienko, P.; Kocon, J.; Kong, J.; Koptyra, B.; Lau, H.; Mantri, K. S. I.; Mom, F.; Saito, A.; Tang, X.; Wang, B.; Wind, J. S.; Wozniak, S.; Zhang, R.; Zhang, Z.; Zhao, Q.; Zhou, P.; Zhu, J.; and Zhu, R.-J. 2023a. RWKV: Reinventing RNNs for the Transformer Era. arXiv:2305.13048.

Peng, B.; Li, C.; He, P.; Galley, M.; and Gao, J. 2023b. Instruction Tuning with GPT-4. arXiv:2304.03277.

Qian, J.; Dong, L.; Shen, Y.; Wei, F.; and Chen, W. 2022. Controllable natural language generation with contrastive prefixes. *arXiv preprint arXiv:2202.13257*.

Qin, C.; Zhang, A.; Zhang, Z.; Chen, J.; Yasunaga, M.; and Yang, D. 2023. Is ChatGPT a General-Purpose Natural Language Processing Task Solver? arXiv:2302.06476.

Qin, L.; Welleck, S.; Khashabi, D.; and Choi, Y. 2022. Cold decoding: Energy-based constrained text generation with langevin dynamics. *Advances in Neural Information Processing Systems*, 35: 9538–9551.

Shu, L.; Luo, L.; Hoskere, J.; Zhu, Y.; Liu, C.; Tong, S.; Chen, J.; and Meng, L. 2023. RewriteLM: An Instruction-Tuned Large Language Model for Text Rewriting. arXiv:2305.15685.

Takase, S.; and Okazaki, N. 2019. Positional Encoding to Control Output Sequence Length. In *Proceedings of NAACL-HLT*, 3999–4004.

Taori, R.; Gulrajani, I.; Zhang, T.; Dubois, Y.; Li, X.; Guestrin, C.; Liang, P.; and Hashimoto, T. B. 2023. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; Rodriguez, A.; Joulin, A.; Grave, E.; and Lample, G. 2023a. LLaMA: Open and Efficient Foundation Language Models. arXiv:2302.13971.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; Bikel, D.; Blecher, L.; Ferrer, C. C.; Chen, M.; Cucurull, G.; Esiobu, D.; Fernandes, J.; Fu, J.; Fu, W.; Fuller, B.; Gao, C.; Goswami, V.; Goyal, N.; Hartshorn, A.; Hosseini, S.; Hou, R.; Inan, H.; Kardas, M.; Kerkez, V.; Khabsa, M.; Kloumann, I.; Korenev, A.; Koura, P. S.; Lachaux, M.-A.; Lavril, T.; Lee, J.; Liskovich, D.; Lu, Y.; Mao, Y.; Martinet, X.; Mihaylov, T.; Mishra, P.; Molybog, I.; Nie, Y.; Poulton, A.; Reizenstein, J.; Rungta, R.; Saladi, K.; Schelten, A.; Silva, R.; Smith, E. M.; Subramanian, R.; Tan, X. E.; Tang, B.; Taylor, R.; Williams, A.; Kuan, J. X.; Xu, P.; Yan, Z.; Zarov, I.; Zhang, Y.; Fan, A.; Kambadur, M.; Narang, S.; Rodriguez, A.; Stojnic, R.; Edunov, S.; and Scialom, T. 2023b. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288.

Wang, Y.; Kordi, Y.; Mishra, S.; Liu, A.; Smith, N. A.; Khashabi, D.; and Hajishirzi, H. 2022. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*.

Wei, J.; Bosma, M.; Zhao, V.; Guu, K.; Yu, A. W.; Lester, B.; Du, N.; Dai, A. M.; and Le, Q. V. 2021. Finetuned Language Models are Zero-Shot Learners. In *International Conference on Learning Representations*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35: 24824–24837.

Xu, B.; Yang, A.; Lin, J.; Wang, Q.; Zhou, C.; Zhang, Y.; and Mao, Z. 2023a. ExpertPrompting: Instructing Large Language Models to be Distinguished Experts. arXiv:2305.14688.

Xu, C.; Sun, Q.; Zheng, K.; Geng, X.; Zhao, P.; Feng, J.; Tao, C.; and Jiang, D. 2023b. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Yang, K.; and Klein, D. 2021. FUDGE: Controlled Text Generation With Future Discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 3511–3535. Online: Association for Computational Linguistics.

Zhang, H.; and Song, D. 2022. DisCup: Discriminator Cooperative Unlikelihood Prompt-tuning for Controllable Text Generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 3392–3406.

Zhang, H.; Song, H.; Li, S.; Zhou, M.; and Song, D. 2022. A survey of controllable text generation using transformer-based pre-trained language models. *arXiv preprint arXiv:2201.05337*.

Zhao, W. X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Zhou, W.; Jiang, Y. E.; Wilcox, E.; Cotterell, R.; and Sachan, M. 2023. Controlled text generation with natural language instructions. *arXiv preprint arXiv:2304.14293*.