Multi-Objective Linguistic Control of Large Language Models

Anonymous ACL submission

Abstract

Large language models (LLMs), despite their breakthroughs on many challenging benchmark tasks, prefer to generate verbose responses and lack the controllability of output complexity, which is usually preferred by human users in practice. In this paper, we study how to precisely control multiple linguistic complexities of LLM output by finetuning using off-the-shelf data. To this end, we propose multi-control tuning (MCTune), which includes multiple linguistic complexity values of ground-truth responses as controls in the input for instruction tuning. We finetune LLaMA2-7B on Alpaca-GPT4 and WizardLM datasets. Evaluations on widely used benchmarks demonstrate that our method does not only improve LLMs' multi-complexity controllability substantially but also retains or even enhances the quality of the responses as a side benefit.

1 Introduction

Large language models have achieved remarkable success in generating free-from texts for different downstream tasks or human instructions. However, existing LLMs still lack precise control over the linguistic complexity of their outputs, e.g., the total number of nouns, the variation of verbs, etc. Linguistic controllability is crucial to creating personalized outputs since those complexity indices directly reflect human reading complexity in multiple aspects. For example, a short yes/no answer is required by some users while a detailed explanation is preferred by others. Moreover, recent studies have discovered a spurious correlation between the quality reward used in LLM alignment and the output length (i.e., a specific linguistic complexity). Consequently, LLMs favor generating verbose responses due to the length bias, which may increase unnecessary reading complexity. It is still an open problem to mitigate the bias without hurting the output quality.

While existing LLM finetuning techniques such as instruction-tuning and reinforcement learning from human feedback (RLHF) has been demonstrated to be effective in aligning the output with human intent or preference, they only focus on maximizing a single objective. Instead, achieving the controllability of multiple complexity indices requires a non-trivial multi-objective optimization that has not been thoroughly studied on LLMs. Rather than solely maximizing or minimizing the complexities, it aims to reach different target complexity values on the Pareto frontier. This requires LLMs to adjust the trade-off among objectives and capture their potential correlations or constraints in the text generation process. In addition, due to the huge space of possible combinations of complexity indices, it could be expensive to collect training data for multi-objective control.

In this paper, we take the first step towards multi-objective control of the linguistic complexity of LLM outputs. Instead of collecting new data, our strategy allows the reuse of existing instruction-tuning data. In particular, we annotate the ground-truth responses in a dataset by their complexity metrics evaluated using tools developed in computational linguistics. The multiple complexity indices and their values are appended as tags to the input so finetuning an LLM on the linguistic-label augmented data helps build a strong connection between the input tags and the linguistic complexity of the output, hence enforcing the LLM to adhere to the complexity requirements during sequential decoding. Surprisingly, we observe that randomly sampling a small subset of tags for each training example suffice to obtain controllability over all the complexities of test examples, thereby reducing the required amount of training data.

We examine our approach by finetuning LLaMA2-7B using a linguistic-complexity labeled Alpaca-GPT4 dataset (i.e., the prompts are from the original Alpaca dataset while the responses

are composed by GPT-4) and a WizardLM dataset. We do not only observe an expected substantial improvement in linguistic controllability but also a side benefit of enhanced response quality, indicating that finetuning under multiple linguistic constraints can improve the LLM for general purposes. Compared to unconstrained RLHF and IFT methods, which suffer from the length bias, our approach does not introduce the bias but can improve both the controllability and quality simultaneously.

2 Related Work

Linguistic Features and Complexity. The exploration of linguistic features and complexity in language models encompasses a diverse range of research. Seminal studies have investigated the syntactic abilities of LSTMs (Linzen et al., 2016). Ficler and Goldberg (2017) has introduced methods for manipulating the stylistics and syntactic output of test generation models. Additionally, linguistic style transfer (Shen et al., 2017) has also showcased the adaptability of language models to capture and replicate varied linguistic features. Building on the previous efforts to understand the multifaceted nature of linguistics complexity, our work concentrates on producing responses endowed with specific linguistic characteristics, which is less explored in the previous work.

Controllability of LLMs. The topic of personalized language modeling has attracted significant attention across various research papers. Techniques such as user embedding have become common for customizing language models to individual needs (Welch et al., 2020; Rocca and Yarkoni, 2022). More recently, Mireshghallah et al. (2022); Oba et al. (2023) propose prompt based personalized fine-tuning for specific users, and producing personalized responses. Our research shifts the focus from personalization for specific users to the broader goal of controlling large language models (LLMs) to produce outputs with linguistic diversity and complexity, addressing a gap not explored by the aforementioned works.

Another prevalent technique for guiding the output of large language models involves Tagging (Korbak et al., 2023; Prabhumoye et al., 2023; Lu et al., 2022). This method incorporates appending human-readable text during the training of LLMs. Contrary to previous studies that concentrated on managing aspects like toxicity (Korbak et al., 2023; Prabhumoye et al., 2023) and controlling repetition (Lu et al., 2022), our approach employs tagging techniques to control linguistic features across multiple attributes. Additionally, we utilize multiple tags to enable simultaneous consideration of various attributes, a difference from earlier work that primarily uses of a single tag.

Finetuning and Alignment of LLMs. With the emergence of large-scale language models, such as those in the GPT series, aligning language models has become prevalent. Studies like those by (Zhou et al., 2023; Xu et al., 2023; Li et al., 2023) have concentrated on the process of data curation for instruction fine-tuning to enhance models' instructions following capabilities. Unlike these data-centric approaches, we keep the original instruction dataset but augment instructions with various tags to introduce a richer array of linguistic features, thereby elevating the instruction-following capabilities of the models.

3 Finetuning LLMs for Linguistic Controllability

In this section, we delineate our approach to multi-control tuning. Section 3.1 outlines the linguistic features of interest and describes how we extract them from a given text segment. In Section 3.2, we explain how the extracted features are incorporated into the multi-control tuning process.

3.1 Handcrafted Linguistic Features

We are specifically interested in controlling the *handcrafted* linguistic properties of the model's generation. This type of feature has been used throughout the NLP field (Bogdanova et al., 2017; Anshika Choudhary, 2021; Lee et al., 2021) and is loosely defined in Lee and Lee (2023) as *"a single numerical value produced by a uniquely identifiable method on any natural language."* An example of a linguistic feature not considered handcrafted is text embeddings produced by deep neural networks, which usually take the form of a vector. We extract such features from a text segment using the LFTK package proposed in Lee and Lee (2023). It encompasses a diverse set of 220 features that are grouped into different linguistic families. Within the scope of this paper, we sample a reasonably-sized set of 14 features for multi-control tuning, which are presented in Table 1. These features are selected to cover most of the feature families while being simple to understand and verify by human users.

ID	Name	Description
1	t_word	number of words
2	n_noun	number of nouns
3	n_verb	number of verbs
4	n_adj	number of adjectives
5	t_uword	number of unique words
6	n_unoun	number of unique nouns
7	n_uverb	number of unique verbs
8	n_uadj	number of unique adjectives
9	ttr	type-token ratio
10	noun_var	noun variation
11	verb_var	verb variation
12	adj_var	adjective variation
13	fkre	Flesch-Kincaid reading ease
14	rt_average	average reading time

Table 1: The list of linguistic features for controllability tuning. The second column shows the name of each feature with the corresponding descriptions in the third column. We include a detailed explanation of how these features are computed in Appendix A.

Formally, given a text segment $\mathbf{x} = [x_1, x_2, ..., x_l]$, with x_i being the i-th token of $\mathbf{x}$, we denote the feature extractor as a function $f : \mathcal{X} \to \mathbb{R}^d$ that maps $\mathbf{x}$ to a d-dimensional vector $f(\mathbf{x}) = [f_1(\mathbf{x}), ..., f_d(\mathbf{x})]$, where $f_j(\mathbf{x})$ represents the j-th linguistic feature of interest and $\mathcal{X}$ represents the space of all texts. In this paper, the function f refers to the LFTK feature extractor.

3.2 Multi-Objective Control Tuning

Consider the standard instruction tuning setting where an LLM, denoted as $p_\theta(\mathbf{x}) = \prod_{i=1}^{l} p_\theta(x_i \mid x_{<i})$, is trained on a dataset of N instruction-output pairs, $D_{\text{train}} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$. The training objective is to generate $\mathbf{y}_i$ given $\mathbf{x}_i$; thus, the loss is

$$\mathcal{L} = -\sum_{i=1}^{N}\sum_{k=1}^{|\mathbf{y}_i|} \log p_\theta(\mathbf{y}_{i,k} \mid \mathbf{x}_i, \mathbf{y}_{i,<k})$$

We address multi-control tuning by framing it as a conditional instruction tuning problem, where we utilize the target response $\mathbf{y}_i$ from D_{train} to generate a linguistic control vector $f(\mathbf{y}_i)$ and append it to $\mathbf{x}_i$. The model is then trained to generate $\mathbf{y}_i$ condition on both $\mathbf{x}_i$ and $f(\mathbf{y}_i)$. However, to enhance data diversity and better simulate real-world scenarios, where a user may wish to control only a few features, we do not utilize all features for every data example. For each pair $(\mathbf{x}_i, \mathbf{y}_i) \in D_{\text{train}}$, we randomly sample an integer $n_i \sim \text{Uniform}\{1, \ldots, m\}$, $m \leq d$. The set of n_i feature indices, denoted by C_i, is then randomly sampled from the pool of all feature indices, i.e., $C_i \sim \text{Uniform}(\{A \subseteq \{1, \ldots, d\} : |A| = n_i\})$. The linguistic control vector used for the i-th example is denoted as $f_{C_i}(\mathbf{y}_i)$, where $f_{C_i}(\mathbf{y}_i) = \left[f_{C_{i,1}}(\mathbf{y}_i), \ldots, f_{C_{i,n_i}}(\mathbf{y}_i)\right]$. The resulting training loss becomes

$$\mathcal{L} = -\sum_{i=1}^{N}\sum_{k=1}^{|\mathbf{y}_i|} \log p_\theta(\mathbf{y}_{i,k} \mid \mathbf{x}_i, f_{C_i}(\mathbf{y}_i), \mathbf{y}_{i,<k})$$

The benefits of our training strategy are twofold: (1) it improves the controllability and instruction-following capability simultaneously, thus avoiding the catastrophic forgetting problem that may degrade the model's generation quality. In fact, we will later show in Section 5.6.2 that LLMs trained with our approach achieve even stronger instruction-following ability compared to those trained with vanilla instruction tuning; (2) it allows us to utilize off-the-shelf datasets without the need to collect new ones.

3.3 Prompt Template

In practice, we format both the instruction $\mathbf{x}$ and the output $\mathbf{y}$ into a predefined template before they are input into the LLMs. This paper adopts the template outlined in (Taori et al., 2023), wherein $\mathbf{x}$ is decomposed into an instruction and an input component. Regarding linguistic controls, we format them into a sequence of the form `[name_1: value_1] ... [name_n: value_n]`, aiming for conciseness by utilizing the features' abbreviated names listed in Table 1. To assist the LLM in better understanding these abbreviations, we provide a comprehensive list of feature descriptions within the system prompt, which has been confirmed to enhance the effectiveness of our approach in preliminary experiments. This sequence of controls is subsequently attached to the input component. Figure 1 illustrates a detailed example of how an input prompt is constructed.

4 Evaluation of Linguistic Controllability

During evaluation on a reserved test dataset D_{test}, we aim to measure how the model performs as the linguistic control vector f changes. To conduct such an evaluation, we develop a sampling strategy to sample different control vectors for an instruction $\mathbf{x}_i$. Specifically, we need a sampling strategy that: (1) is specific to an instruction $\mathbf{x}_i$, i.e., the

Below is an instruction that describes a task, paired with an input that provides further context. At the end of the input, there will be a list of tags specifying the desired properties of the response. The following tags are available:
[t_word] for the total number of words; [n_noun] for the total number of nouns;
[n_verb] for the total number of verbs; [n_adj] for the total number of adjectives;
[t_uword] for the total number of unique words; [n_unoun] for the total number of unique nouns;
[n_uverb] for the total number of unique verbs; [n_uadj] for the total number of unique adjectives;
[simp_ttr] for the simple type-token ratio; [simp_noun_var] for simple noun variation;
[simp_verb_var] for simple verb variation; [simp_adj_var] for simple adjective variation;
[fkre] for the Flesch-Kincaid Reading Ease; [rt_average] for the average reading time.
Write a response that appropriately completes the request and satisfies the tags.
Instruction:
Arrange the words in the given sentence to form a grammatically correct sentence.
Input:
quickly the brown fox jumped `[t_word: 6] [n_noun: 1] [fkre: 102.05]`
Response:
The brown fox jumped quickly.

Figure 1: An example of how data is formatted before being fed into LLMs in this paper. The first paragraph presents a system prompt containing a complete list of feature descriptions. Our preliminary results indicate that including descriptions enhances the effectiveness of our approach.

sampled control vectors should not stray too far from the reasonable range for a specific $\mathbf{x}_i$. For example, an instruction to "Generate a short story" should not have a large value for `t_word`; (2) ensures the sampled control vectors are always valid, meaning that no linguistic controls conflict with each other (e.g., `t_uword` should always be less than or equal to `t_word`), and no control is out-of-bound (e.g., `fkre` should always be less than or equal to 121.22).

To achieve the first goal, we utilize $\mathbf{y}_i$'s linguistic feature vector as a reference point and sample new control vectors f' from the Gaussian distribution centered at $f(\mathbf{y}_i)$, i.e., $f' \sim \mathcal{N}(f(\mathbf{y}_i), I\sigma^2)$. However, since each feature $f_i(\mathbf{y})$ has a different range, a small σ for some features may be large for others. To avoid this inconsistency, we standardize all features to unit variance before sampling. More formally, the new control vector f' is computed by

$$f' = z^{-1}(z(f(\mathbf{y}_i)) + \sigma\epsilon), \quad \epsilon \sim \mathcal{N}(0, I)$$

where $z : \mathbb{R}^d \to \mathbb{R}^d$ standardizes each feature to unit variance, and z^{-1} is the inverse operation.

While sampling a valid control vector can be a challenging problem, verifying its validity is straightforward. This can be done using a simple rule-based method, which we outline in Appendix B. We utilize this observation to achieve the second goal by performing rejection-based sampling, i.e., keep resampling a new control vector f' until we find a valid one. We will show in Section 5.7.3 that σ can serve as a hyperparameter to control the evaluation difficulty.

Lastly, for each example in D_{test}, we randomly sample a number n, a set of n feature indices C, and K new control vectors f'_C for controllability and generation quality evaluation.

5 Experiments

5.1 Implementation Details

By default, we set $K = 5$ and $\sigma = 0.1$ unless specified otherwise. We set the maximum number of linguistic controls per example to $m = 5$ and will show in Section 5.7.2 that limiting m to 5 does not affect the model's controllability, even when more than 5 controls are used in the evaluation. For all datasets, we fine-tune LLaMA2-7B using the AdamW optimizer with a linear learning rate schedule. We set the learning rate to 2×10^{-5}, the batch size to 128, and the number of warmup steps to 100. All models are trained on 8 RTX A6000 GPUs for 5 epochs.

5.2 Datasets.

Data Preprocessing. Because LFTK cannot extract linguistic features with perfect accuracy, it sometimes generates invalid control vectors $f(\mathbf{y}_i)$ (e.g., producing an `fkre` greater than 121.22). This situation complicates the process of sampling new control vectors, as $f(\mathbf{y}_i)$ may deviate significantly from the feasible region, drastically reducing the probability of sampling a valid one. Therefore, we exclude data examples with invalid $f(\mathbf{y}_i)$.

Alpaca-GPT4. We utilize the Alpaca-GPT4 dataset (Peng et al., 2023) for instruction tuning and evaluation. This dataset shares the same set of instructions as the Alpaca dataset (Taori et al., 2023), but it employs OpenAI's GPT-4 to generate high-quality responses. Our preliminary experiments demonstrate that training with this dataset yields better results in terms of both controllability and generation quality compared to training with the original Alpaca dataset. After preprocessing, the dataset is divided into a training set of 45,000 examples and a test set of 2,000 examples.

WizardLM. In addition to Alpaca-GPT4, we also evaluate our method on the WizardLM dataset (Xu et al., 2023). The original dataset contains 70,000 examples of instruction-output pairs that were automatically generated by ChatGPT. We subsampled 50,000 examples from the original dataset. Similar to Alpaca-GPT4, after preprocessing, we split the data into a training set of 40,000 examples and a test set of 2,000 examples.

5.3 Models.

We use LLaMA2-7B (Touvron et al., 2023) as the base model for multi-control tuning in all experiments.

5.4 Baselines

To evaluate the impact of our method on controllability and generation quality, we conduct comparisons with the same LLaMA2-7B base model but trained with regular instruction fine-tuning. We also include OpenAI's GPT-3.5 Turbo (gpt-3.5-turbo-0125) as a baseline to compare our model against state-of-the-art proprietary LLMs in terms of controllability. For each baseline, we control the linguistic complexity of their responses using the prompt template shown in Figure 1 in a zero-shot manner.

5.5 Evaluation Metrics

Controllability Error. Given an instruction $\mathbf{x}$, a linguistic control vector f_C, and a generated response $\hat{\mathbf{y}} \sim p_\theta(\mathbf{y} \mid \mathbf{x}, f_C)$, we measure controllability error by computing the L_1 error between the specified control vector f_C and the response's linguistic feature vector $f_C(\hat{\mathbf{y}})$. We denote this error as $\mathbf{e} = |f_C(\hat{\mathbf{y}}) - f_C| \in \mathbb{R}^{|C|}$.

Quality Score. To evaluate generation quality, we follow Zheng et al. (2023) in using a powerful LLM (e.g., ChatGPT-4 Turbo) as a judge to assign quality scores ranging from 1 to 10.

5.6 Main Results and Analysis

5.6.1 Linguistic Controllability Evaluation

This section presents the evaluation of linguistic controllability between our method and various baselines. We consider three evaluation settings: Easy ($\sigma = 0.1$), Medium ($\sigma = 0.2$), and Hard ($\sigma = 0.3$). Our goal is to measure the controllability error of each baseline on each linguistic control and then visualize all of them on the same radar plot for comparison. Formally, for each linguistic control i, and each baseline j, we maintain a list $\mathbf{e}_{i,j} = \left[e^1_{i,j}, \ldots, e^{|\mathbf{e}_{i,j}|}_{i,j}\right]$ of L_1 errors made by j, where the length of this list depends on the number of text examples that contain i. The matrix of all L_1 errors for feature i is denoted as

$$\mathbf{E}_i = \begin{pmatrix} e^1_{i,1} & \cdots & e^{|\mathbf{e}_{i,1}|}_{i,1} \\ \vdots & \ddots & \vdots \\ e^1_{i,J} & \cdots & e^{|\mathbf{e}_{i,J}|}_{i,J} \end{pmatrix} \quad (1)$$

It is challenging to directly visualize $\mathbf{E}_i$ for all i on the same radar plot because each linguistic control i has a different range of values. Therefore, we normalize each element in $\mathbf{E}_i$ using min-max normalization, with the minimum being $\min E_i$ and the maximum being the 95th percentile of $\mathbf{E}_i$, or $P_{95}(\mathbf{E}_i)$. For each baseline j, its average normalized L_1 error on control i is calculated as $\sum_k \text{norm}(e^k_{i,j})/|\mathbf{e}_{i,j}|$, where $\text{norm}(\cdot)$ denotes the normalization described above.

As shown in Figure 2, our method consistently outperforms the other baselines across all settings. Surprisingly, a state-of-the-art model like ChatGPT underperforms in controllability compared to the instruction-finetuned LLaMA2-7B. Upon manual inspection, we observed that ChatGPT tends to produce more verbose responses regardless of the linguistic controls applied, which may explain the observed poor performance. Another possible reason for this discrepancy is that LLaMA2-7B is finetuned on data sharing the same distribution as the test set, giving it an advantage over ChatGPT. Note that for `noun_var`, `verb_var`, and `adj_var`, the differences between each baseline's errors are not significant. We hypothesize that, in contrast to other linguistic complexities, the descriptions of these features are somewhat more ambiguous, and we also did not provide a clear description in the

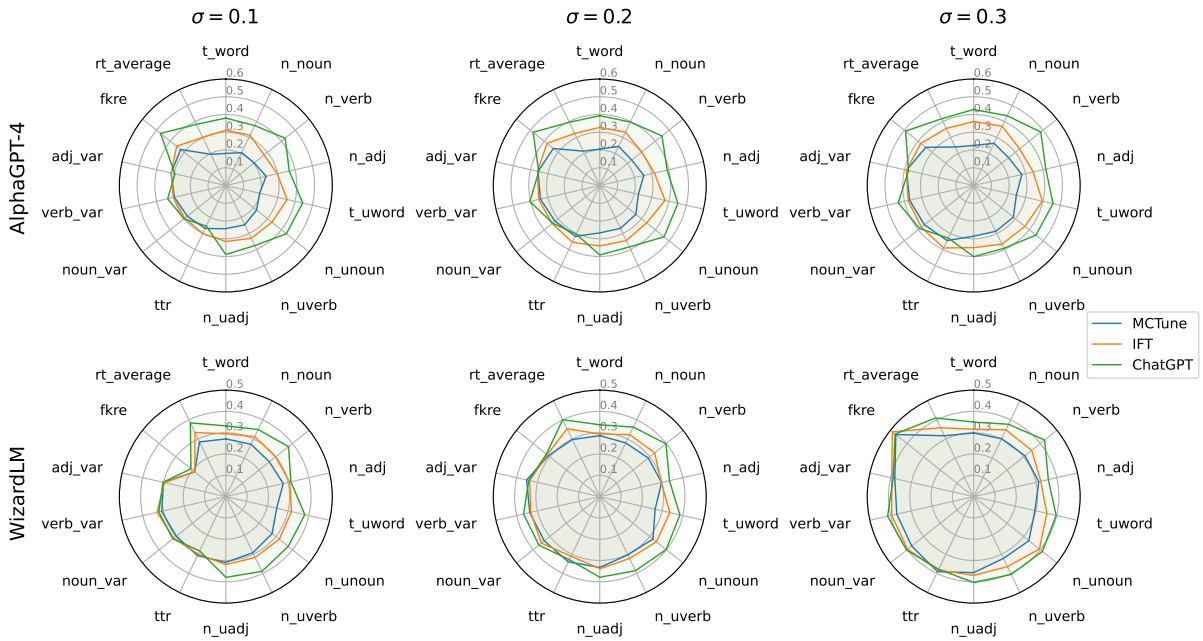

Figure 2: **Comparison of controllability error** (the average normalized L_1 error) of ChatGPT (`gpt-3.5-turbo-0125`), IFT (finetuning without controls), and MCTune (ours) on AlpacaGPT-4 and WizardLM datasets and three test settings of target linguistic complexity with increasing σ (difficulty). To visualize each linguistic feature's average error on the same radar plot, we apply min-max normalization to normalize them to a similar scale. To reduce the effect of outliers, the minimum L_1 error of the i-th feature is the minimum among all baselines, and the maximum L_1 error refers to the 95th percentile of errors among all baselines.

system prompt, which may not be helpful for the model to understand and follow these features.

5.6.2 Generation Quality Evaluation

A natural concern when fine-tuning for controllability is how it affects the model's generation quality. In this section, we answer this question by comparing our method with standard instruction fine-tuning on MT-Bench (Zheng et al., 2023), a widely used benchmark for evaluating LLMs on multi-turn open-ended questions. Specifically, we use GPT-4 Turbo (`gpt-4-0125-preview`) as a judge and compare the two methods in both single-answer and pairwise settings. In the single-answer setting, the judge assigns a single numeric score from 1 to 10 for each model's answer. In the pairwise setting, the judge receives two answers from both baselines and returns either a win, a loss, or a tie. The result for the single-answer setting is shown in Figure 4. As shown in the figure, our method does not degrade the model's general language ability but even improves it in most categories. Note that the performance of both baselines on Coding and Math questions is poor because the AlpacaGPT-4 dataset does not have a strong specialty in questions of this category. When evaluating under the pairwise setting, MCTune achieves a 51.25% win rate over instruction fine-tuning. These results suggest that training for controllability is beneficial for improving LLMs' natural language performance.

5.7 Analyses

5.7.1 Relationship Between Linguistic Controllability and Generation Quality

This experiment focuses on studying the relationship between generation quality and the controllability error of LLM responses. To start off, we randomly select a set of 5 linguistic controls C and fix it throughout the experiment for simplicity. Given an LLM p_θ and the Alpaca-GPT4 test dataset $D_{\text{test}} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$, we select 50 examples from D_{test} then sample 50 responses $\hat{\mathbf{y}}_i$ from $p_\theta(\mathbf{y}_i \mid \mathbf{x}_i, f_C(\mathbf{y}_i))$. Each $\hat{\mathbf{y}}_i$ is then evaluated by GPT-4 Turbo on how well it satisfies $\mathbf{x}_i$ and a controllability error of $\hat{\mathbf{y}}_i$ is computed by taking the average of the normalized L_1 errors across all controls in C. We also consider three different settings where $\sigma = 0.1$, $\sigma = 0.3$, and $\sigma = 0.5$ to see how the pattern shifts as σ changes. The results are shown in Figure 3. We can see that, compared to

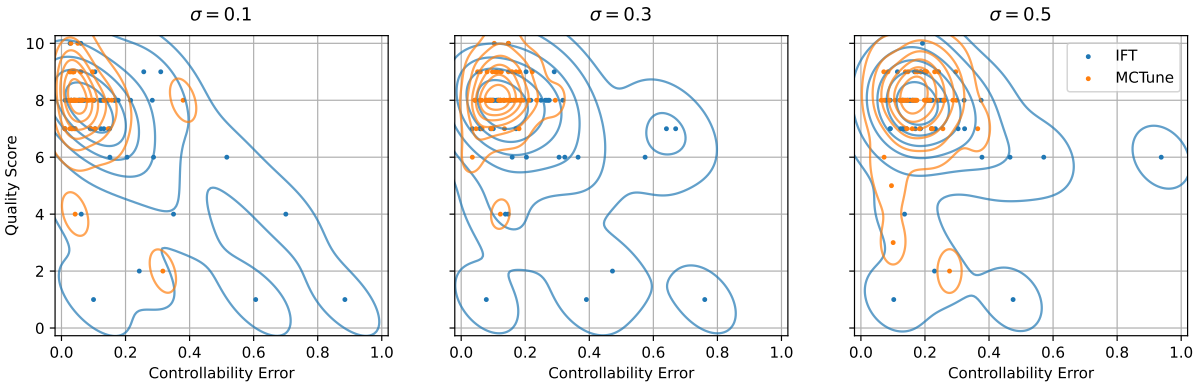

Figure 3: **Trade-off between linguistic controllability and generation quality** in three increasingly difficult test settings. Each dot represents a model's response $\hat{\mathbf{y}}$ to a specific query $[\mathbf{x}, f_C]$. The response is given a quality score from 1 to 10 by a judge LLM (GPT-4 Turbo) based on how well $\hat{\mathbf{y}}$ addresses $\mathbf{x}$. A controllability error is measured for $\hat{\mathbf{y}}$, which is computed by taking the average of normalized L_1 errors across all linguistic controls in f_C. Blue and orange dots respectively represent responses from models trained by IFT and MCTune using Alpaca-GPT4 dataset.

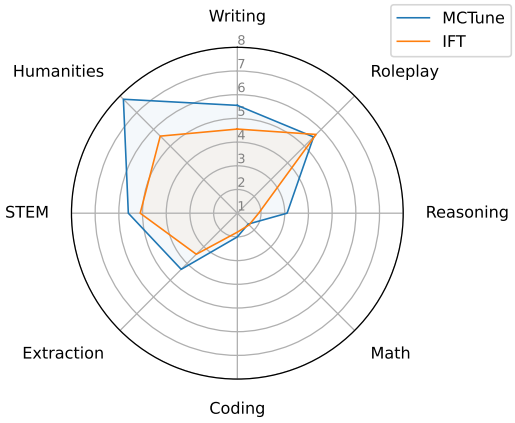

Figure 4: **Comparison between MCTune and IFT-trained models on MT-Bench.** We finetune LLaMA2-7B on Alpaca-GPT4 dataset and GPT-4 Turbo is the judge in the test. The average score per axis ranges from 1 to 10 and are given by the judge.

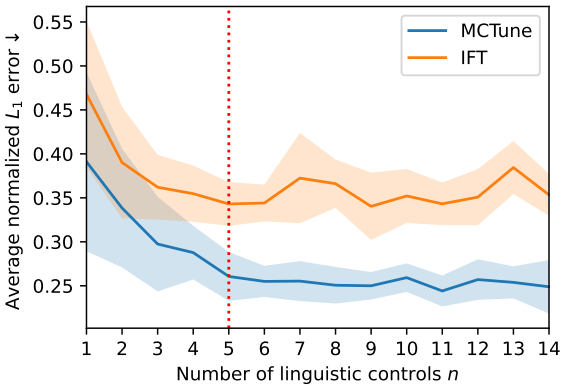

Figure 5: Linguistic controllability error on the test set as the number of linguistic controls n per sample in MCTune's training increases. The solid curves represent the linguistic controllability error averaged over all linguistic complexities (lower is better) with the shaded areas represent the 95%-confidence interval. The dotted vertical line indicates the maximum number of controls (5) used during MCTune's training.

IFT, our method achieves better quality and controllability simultaneously. As σ increases, the controllability error increases, which is expected and consistent with the results in Section 6. We then start to notice a slight degradation in quality at $\sigma = 0.5$, where there are no responses with a score of 10, and more responses with low scores begin to appear. This observation suggests that there might be a positive correlation between controllability and generation quality.

5.7.2 Analysis of Model Controllability with Varying Linguistic Controls

In this experiment, we are interested in studying how the model's controllability is affected as the number of linguistic controls increases. We conduct this experiment using the Alpaca-GPT4 dataset, from which we randomly sample 100 examples from the test set. For each example, we sample five new control vectors f' of length n, compute the average normalized L_1 error per control, and then average these over all controls to

obtain a single numeric score. We repeat this process for $n = 1$ to $n = 14$. The results are shown in Figure 5. Surprisingly, controllability is worse when n is small. This is true for both MCTune and the IFT baseline, suggesting that this is not an inherent limitation of our approach. Notice that, as the number of linguistic controls increases beyond five, the maximum number of controls used during training, the controllability error of IFT fluctuates, while MCTune remains approximately stable. This demonstrates that our model, trained with MCTune, is able to generalize to a larger number of controls than it has seen in training.

5.7.3 Analyzing the Impact of σ on Linguistic Controllability Evaluation

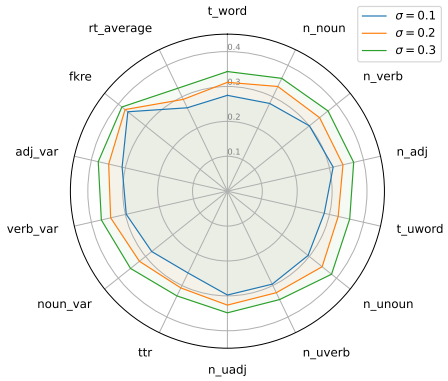

Figure 6: Analysis of how the average normalized L_1 error varies when **changing the test difficulty level** via σ. We use the same model and only vary σ.

This section examines the effect of σ on controllability evaluation. Given a model fine-tuned with MCTune and a test dataset, we follow the process described in Section 4 to sample new linguistic control vectors for evaluating controllability. We consider three settings: $\sigma = 0.1$, $\sigma = 0.2$, and $\sigma = 0.3$. The results are depicted in Figure 6. As illustrated by the results, as σ increases, the average normalized L_1 error uniformly increases across all linguistic controls. This indicates that σ can serve as a hyperparameter to adjust the difficulty of the evaluation, which is intuitive since a larger σ enables the sampling of control vectors that are more distant from the reference control vector $f(\mathbf{y})$.

5.8 Case Studies

This section presents examples of how different values of controls affect the model's responses. As shown in Figure 7, increasing the `t_word` control leads to a longer response and larger values of `n_verb` and `n_uverb` increase the variation in verb choice. Although it is challenging to strictly control the complexities of the response, the linguistic controls can serve as soft constraints on the model's generation.

Instruction: Given an economic concept, define it in simple terms.
Input: Inflation `[t_word: 54] [n_verb: 9] [n_uverb: 8]`
Response: Inflation is when the prices of goods and services increase over time. This means that a dollar will buy less in the store as the year goes by. While some inflation is expected to happen naturally due to growth and increased demand, high inflation rates can harm the economy and cause uncertainty.

Response's complexities:
`t_word=56, n_verb=9, n_uverb=8.`

Instruction: Given an economic concept, define it in simple terms.
Input: Inflation `[t_word: 109] [n_verb: 15] [n_uverb: 14]`
Response: Inflation is when the prices of goods and services rise over time. Imagine a dollar bill losing value over time, where a single dollar can't buy as much as it used to. That's inflation. It's important to note that inflation is not the same as increasing prices on purpose, which is called deflation. Inflation is generally considered a positive thing because it means the economy is growing and people have more money to spend. However, high inflation can also lead to rising prices for essentials like food and housing, which can be a challenge for people on a fixed income.

Response's complexities:
`t_word=113, n_verb=16, n_uverb=15.`

Figure 7: Examples of MCTune model responding to the same instruction under different linguistic controls. The linguistic complexities of the responses are extracted using LFTK with exact values shown at the bottom of each example.

6 Discussion and Conclusions

In this paper, we advance the precise control of linguistic complexities in LLMs through multi-objective control tuning. Our method is straightforward and can be seamlessly integrated with existing instruction-tuning datasets without the need to gather new ones. Our training objective concurrently optimizes the LLM's ability to follow instructions and its controllability. Through our experiments, we find that incorporating this dual-focus strategy significantly improves the LLM's generative quality, surpassing the results of instruction fine-tuning alone. This finding suggests that controllability and instruction-following ability may have a complementary effect on each other. Additionally, we observe that while state-of-the-art LLMs achieve impressive natural language performance, they are not easy to control. This emphasizes the need for studying methods that improve controllability in LLMs.

7 Limitations

While our method significantly improves controllability compared to regular instruction fine-tuning, we observe that there is still room for improvement. The model trained with MCTune is able to loosely follow the linguistic controls but struggles to produce responses with the exact complexities. An interesting next step would be to improve controllability in a strict setting. Another limitation we observe is the fact that LFTK sometimes extracts incorrect linguistic complexities for a given text. This leads to noisy controls that may confuse and reduce the controllability of the model during training.

References

Anuja Arora Anshika Choudhary. 2021. Linguistic feature based learning model for fake news detection and classification. *Expert Systems with Applications*.

Dasha Bogdanova, Jennifer Foster, Daria Dzendzik, and Qun Liu. 2017. If you can't beat them join them: Handcrafted features complement neural nets for non-factoid answer reranking. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 121–131, Valencia, Spain. Association for Computational Linguistics.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. In *Proceedings of the Workshop on Stylistic Variation*, pages 94–104, Copenhagen, Denmark. Association for Computational Linguistics.

Tomasz Korbak, Kejian Shi, Angelica Chen, Rasika Vinayak Bhalerao, Christopher Buckley, Jason Phang, Samuel R. Bowman, and Ethan Perez. 2023. Pretraining language models with human preferences. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 17506–17533. PMLR.

Bruce W. Lee, Yoo Sung Jang, and Jason Hyung-Jong Lee. 2021. Pushing on text readability assessment: A transformer meets handcrafted linguistic features.

Bruce W. Lee and Jason Lee. 2023. LFTK: Handcrafted features in computational linguistics. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 1–19, Toronto, Canada. Association for Computational Linguistics.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2023. From quantity to quality: Boosting llm performance with self-guided data selection for instruction tuning. *ArXiv*, abs/2308.12032.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of lstms to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Ximing Lu, Sean Welleck, Jack Hessel, Liwei Jiang, Lianhui Qin, Peter West, Prithviraj Ammanabrolu, and Yejin Choi. 2022. Quark: Controllable text generation with reinforced unlearning. *Advances in neural information processing systems*, 35:27591–27609.

Fatemehsadat Mireshghallah, Vaishnavi Shrivastava, Milad Shokouhi, Taylor Berg-Kirkpatrick, Robert Sim, and Dimitrios Dimitriadis. 2022. UserIdentifier: Implicit user representations for simple and effective personalized sentiment analysis. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3449–3456, Seattle, United States. Association for Computational Linguistics.

Daisuke Oba, Naoki Yoshinaga, and Masashi Toyoda. 2023. Perplm: Personalized fine-tuning of pretrained language models via writer-specific intermediate learning and prompts. *arXiv preprint arXiv:2309.07727*.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Shrimai Prabhumoye, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. 2023. Adding instructions during pretraining: Effective way of controlling toxicity in language models. *arXiv preprint arXiv:2302.07388*.

Roberta Rocca and Tal Yarkoni. 2022. Language as a fingerprint: Self-supervised learning of user encodings using transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1701–1714, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. *Advances in neural information processing systems*, 30.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa,

Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Charles Welch, Jonathan K. Kummerfeld, Verónica Pérez-Rosas, and Rada Mihalcea. 2020. Exploring the value of personalized word embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6856–6862, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2023. Lima: Less is more for alignment. *arXiv preprint arXiv:2305.11206*.

A Computing Linguistic Features

In this section, we describe in greater detail how linguistic features are computed. We use LFTK as a feature extractor, which is based on spaCy. Given a string, spaCy tokenizes it into a sequence of tokens along with their annotations (e.g., part-of-speech). The number of words `t_word` is the number of tokens. The `n_noun`, `n_verb`, and `n_adj` are computed based on the POS provided by spaCy. The `t_uword`, `n_unoun`, `n_uverb`, and `n_uadj` are computed accordingly. The type-token ratio, noun variation, verb variation, and adjective variation are $\frac{\texttt{t\_uword}}{\texttt{t\_word}}$, $\frac{\texttt{n\_unoun}}{\texttt{n\_noun}}$, $\frac{\texttt{n\_uverb}}{\texttt{n\_verb}}$, and $\frac{\texttt{n\_uadj}}{\texttt{n\_adj}}$, respectively. Let `t_sent` be the number of sentences and `t_syll` is the number of syllables, the formula to compute Flesch-Kincaid reading ease is

$$206.835 - 1.015\left(\frac{\texttt{t\_word}}{\texttt{t\_sent}}\right) - 84.6\left(\frac{\texttt{t\_syll}}{\texttt{t\_word}}\right)$$

Lastly, the average reading time `rt_average` is $\frac{\texttt{t\_word}}{240}$.

B Verifying Validity of Control Vectors

Given a linguistic control vector, it is straightforward to check its validity by iterating through a list of pre-defined rules.

- $\texttt{t\_word} > 0$
- $\texttt{t\_word} \geq \texttt{n\_noun} + \texttt{n\_verb} + \texttt{n\_adj}$
- $\texttt{t\_word} \geq \texttt{t\_uword}$
- $\texttt{n\_noun} \geq 0$
- $\texttt{n\_noun} \geq \texttt{n\_unoun}$
- $\texttt{n\_verb} \geq 0$
- $\texttt{n\_verb} \geq \texttt{n\_uverb}$
- $\texttt{n\_adj} \geq 0$
- $\texttt{n\_adj} \geq \texttt{n\_uadj}$
- $\texttt{t\_uword} > 0$
- $\texttt{t\_uword} \geq \texttt{n\_unoun} + \texttt{n\_uverb} + \texttt{n\_uadj}$
- $\texttt{n\_unoun} \geq 0$
- $\texttt{n\_uverb} \geq 0$
- $\texttt{n\_uadj} \geq 0$
- $\texttt{fkre} \leq 121.22$

If any of the rules above is violated, we conclude that the control vector is invalid.