Controllable Style Arithmetic with Language Models

Weiqi Wang[1], **Wengang Zhou**[1*], **Zongmeng Zhang**[1], **Jie Zhao**[2], **Houqiang Li**[1*]
[1]University of Science and Technology of China, [2]Huawei
wangweiqi0329@mail.ustc.edu.cn, zhwg@ustc.edu.cn,
zhangzm@mail.ustc.edu.cn, zhaojie104@huawei.com, lihq@ustc.edu.cn

Abstract

Language models have shown remarkable capabilities in text generation, but precisely controlling their linguistic style remains challenging. Existing methods either lack fine-grained control, require extensive computation, or introduce significant latency. We propose Style Arithmetic (SA), a novel parameter-space approach that first extracts style-specific representations by analyzing parameter differences between models trained on contrasting styles, then incorporates these representations into a base model with precise control over style intensity. Our experiments show that SA achieves three key capabilities: controllability for precise adjustment of styles, transferability for effective style transfer across tasks, and composability for simultaneous control of multiple style dimensions. Compared to alternative methods, SA offers superior effectiveness while achieving optimal computational efficiency. Our approach opens new possibilities for flexible and efficient style control in language models.

1 Introduction

Language models have demonstrated remarkable capabilities in generating human-like text across various tasks (OpenAI et al., 2024). However, it remains a significant challenge to efficiently control the linguistic style of their outputs with fine-grained granularity. The ability to modulate styles, such as conciseness and readability, is crucial for applications ranging from personalized AI assistants to adaptive educational systems, especially given the models' versatility in handling different types of tasks.

Researchers have proposed various approaches to linguistic style control, including prompt engineering (Chen and Moscholios, 2024), fine-tuning (Liu et al., 2024), and collaborative decoding (Shi et al., 2024) methods. However, prompt engineering only allows for coarse-grained style control without precise adjustment; model fine-tuning requires separate training for each style-task combination, resulting in substantial computational overhead; and collaborative decoding methods necessitate multiple forward passes during inference, leading to system latency. Furthermore, previous work has primarily focused on conversational tasks, leaving the style transfer across different functionali-

*Corresponding authors

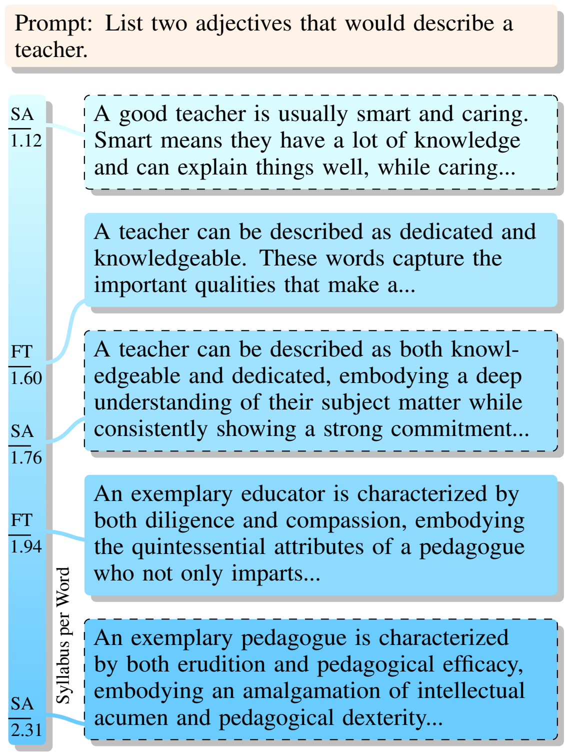


Figure 1: The visualization presents five responses to a prompt. The bars on the left indicate the average number of syllables per word for each response—a metric that reflects vocabulary difficulty—where darker shades correspond to higher values. Two responses marked as FT are generated by models fine-tuned on basic and advanced training data respectively, while the other three marked as SA are produced using our Style Arithmetic method.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 15750–15799
July 27 - August 1, 2025

ties of modern general-purpose language models largely unexplored.

Inspired by recent advances in model merging techniques that effectively combine capabilities from multiple models (Zhou et al., 2024; Yang et al., 2024b), we propose Style Arithmetic (SA), a novel approach that operates directly in the parameter space. SA first extracts style-specific representations by analyzing the parameter differences between models trained on contrasting styles, then incorporates these representations into a base model with precise control over style intensity.

Figure 1 demonstrates SA's remarkable capabilities in controlling vocabulary difficulty. Unlike fine-tuning (FT) approaches that can only produce responses at fixed difficulty levels matching the training data, SA enables flexible interpolation, generating text with difficulty levels that smoothly vary between training levels, as well as extrapolation, producing text that extends beyond training levels to be either simpler than the basic training data or more sophisticated than the advanced training data.

As a parameter-space approach, our SA brings two key advantages: First, it achieves precise and efficient control over linguistic styles through simple arithmetic operations on model parameters. Second, it enables effective style transfer across different tasks, as the extracted style representations capture generalizable stylistic features rather than task-specific patterns.

To demonstrate the effectiveness of SA, we first conduct experiments to evaluate its style **controllability** by showing how extracted stylistic representations can precisely modulate model outputs. We then extend this investigation to demonstrate that these representations enable cross-task style **transferability**. Our experiments further reveal the **composability** of style representations, showing that multiple style representations can be effectively combined, enabling sophisticated style control. Through comprehensive comparisons with baselines including prompting, mixed supervised fine-tuning and collaborative decoding, we demonstrate that SA achieves superior performance in both effectiveness and efficiency.

2 Related Work

2.1 Model Collaboration

Language models have evolved to excel in different domains and exhibit diverse linguistic styles. Model collaboration leverages these differences to enhance both task performance and style control by integrating multiple models.

Two main approaches exist for model collaboration. Pre-inference methods include parameter averaging (Liao et al., 2024) and task vectors (Ilharco et al., 2023). While direct averaging can be suboptimal (Yang et al., 2024a), weighted averaging with optimized coefficients shows promise (Zhou et al., 2024; Goddard et al., 2024). Task vectors enable capabilities like multitask learning and controlled forgetting through parameter arithmetic. During inference, models can collaborate by merging probability distributions (Hoang et al., 2024; Li et al., 2024).

2.2 Linguistic Style Control

Linguistic style control focuses on generating responses with specific linguistic characteristics, encompassing related areas such as personalization (Zhang et al., 2024), role playing (Chen et al., 2024a), and controlled text generation (Liang et al., 2024). Training-free approaches include prompt engineering (Ge et al., 2023), agents (Chen et al., 2024b) and composing or biasing existing LLMs Dekoninck et al. (2024).

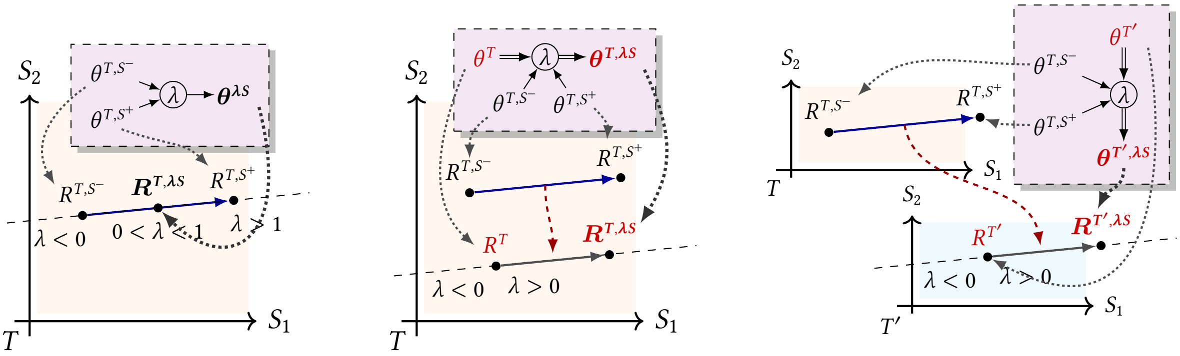
Training-based methods achieve more precise style control by directly optimizing model parameters. These methods primarily include activation manipulation (Konen et al., 2024), fine-tuning (Nguyen et al., 2024) and reinforcement learning approaches (Xu et al., 2024; Ramé et al., 2023). Notably, Ramé et al. (2023) explored parameter averaging (Liao et al., 2024) in the context of multi-objective RLHF, demonstrating that model weights maintain linear relationships when fine-tuned on different rewards from a shared initialization. In contrast, we investigate task vectors (Ilharco et al., 2023) for style control, enabling a wide range of linguistic style adjustments and transfers across various tasks - an approach that, to our best knowledge, has not been previously explored.

3 Methodology

We formalize the style control problem and introduce both baselines and our method in this section. See Appendix A for more details.

3.1 Problem Formulation

We begin by defining two concepts, *i.e.,* task and style. A **task** T is characterized by two compo-



(a) Controllability: $\theta^{T,S^\pm}$ have the same task but opposite styles

(b) Transferability: θ^T has the same task with $\theta^{T,S^\pm}$ but arbitrary style

(c) Transferability: $\theta^{T'}$ has different task with $\theta^{T,S^\pm}$ and arbitrary style

Figure 2: To illustrate our goals, we establish a coordinate system that represents the styles of a model θ and its responses R. Their coordinates are determined by the style metrics $\left(M^{S_1}(R), M^{S_2}(R)\right)$. $\theta^{T,\lambda S}$ represents a language model that performs task T while exhibiting style λS. The dashed boxes represent the construction of $\theta^{\lambda S}$. The gray dashed arrows indicate the correspondence between models and their responses, and the red dashed arrows illustrate the style transfer process.

nents: a test instruction set I^T_{Test} and a task performance metric M^T.

A **style**, formally denoted as λS, consists of two aspects: the style dimension S which represents the specific linguistic feature we aim to control, such as response length or vocabulary difficulty, and the style intensity λ which indicates the degree of that feature's presence in the response. A style dimension S consists of a pair of contrasting adjectives $S^\pm$ representing opposite ends of the style spectrum, and a style metric M^S that quantifies the style characteristics in responses. For example, response length can be characterized by the endpoints "concise" (S^-) and "verbose" (S^+), with token count serving as its metric M^S.

Based on this definition, we can express a style λS as a linear combination of its endpoints: $\lambda S = (1-\lambda) S^- + \lambda S^+$. This formulation naturally extends to multiple style dimensions. For instance, given two style dimensions S_1 and S_2 with corresponding intensities λ_1 and λ_2, their combination $\lambda_1 S_1 + \lambda_2 S_2$ also constitutes a valid style.

Our primary goal is illustrated in Figure 2a. Given two models $\theta^{T,S^\pm}$ that generate responses $R^{T,S^\pm}$ for task T with contrasting styles $S^\pm$, we aim to create a system λ that produces $\theta^{T,\lambda S}$ generating responses $R^{T,\lambda S}$ lying along the line connecting the style endpoints, $R^{T,S^\pm}$. The relationship between $R^{T,\lambda S}$ and $R^{T,S^\pm}$ can be formally expressed as:

$$M^S\left(R^{T,\lambda S}\right) \approx (1-\lambda) M^S\left(R^{T,S^-}\right) + \lambda M^S\left(R^{T,S^+}\right) \tag{1}$$

$$= \underbrace{M^S\left(R^{T,S^-}\right)}_{\text{Initial Point}} + \lambda \underbrace{\left[M^S\left(R^{T,S^+}\right) - M^S\left(R^{T,S^-}\right)\right]}_{\text{Controlling Direction}}. \tag{2}$$

For $0 < \lambda < 1$, $R^{T,\lambda S}$ represents an interpolation between $R^{T,S^\pm}$, which we aim to make as smooth as possible. When $\lambda < 0$ or $\lambda > 1$, $R^{T,\lambda S}$ represents an extrapolation beyond the original style endpoints. We refer to this goal as **controllability**.

As in controlled text generation, $\theta^{T,\lambda S}$ must not only generate responses with the desired style characteristics but also maintain high response quality (Liang et al., 2024). Many factors can affect response quality, and style may be one of them, as shown by differences in task performance $M^T\left(R^{T,S^\pm}\right)$ between style endpoints. For instance, in mathematical reasoning, overly concise responses that omit intermediate steps lead to incorrect solutions.

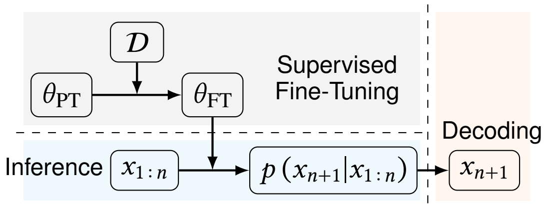
For interpolation, since $M^T\left(R^{T,S^\pm}\right)$ reflects the inherent capabilities of the models rather than being method-dependent, we require that

$$M^T\left(R^{T,\lambda S}\right) \geq (1-\lambda) M^T\left(R^{T,S^-}\right) + \lambda M^T\left(R^{T,S^+}\right) \tag{3}$$

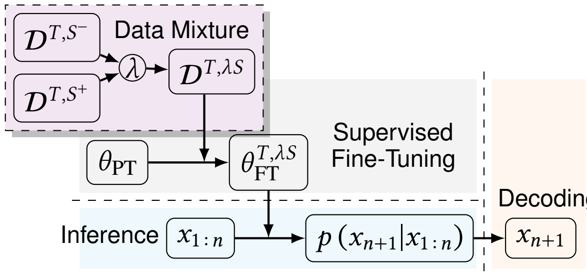
to ensure the model maintains high response quality. For extrapolation, while we acknowledge that some performance degradation may be inherent to style intensification, we allow for gradual performance decline as style intensifies but prohibit cliff-like degradation caused by model collapse.

Beyond controllability, we introduce another goal called style **transferability**. Eq. (2) suggests that we can treat any model as a initial point and use $\theta^{T,S^\pm}$ as reference points to adjust its style. This means we can relax two key constraints:

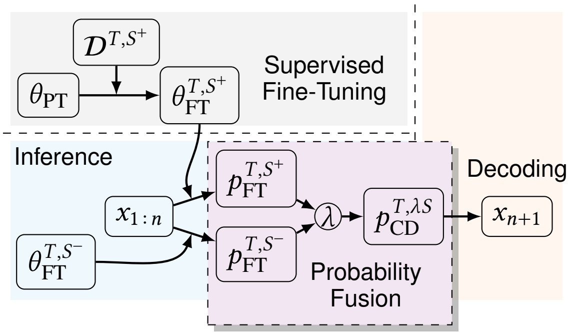
First, the initial model does not need to exhibit style S^-. Moving from Figure 2a to 2b, we relax



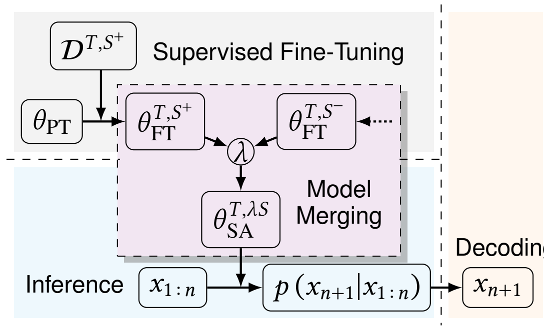
(a) Training and deployment pipeline for language models, with the autoregressive process omitted for simplicity.



(b) MSFT method (baseline 2): Training data with contrasting styles are mixed before training. For each choice of λ, a separate training process is required.



(c) CD method (baseline 3): Probability distributions are fused after inference. There is no overhead when selecting λ, however, an extra inference pass is required for each input instruction.



(d) **SA method**: Models are merged in parameter space after training. For each choice of λ, a merging operation is required.

Figure 3: Illustration of different methods at various levels. θ_{PT} and θ_{FT} represent the pretrained and fine-tuned models, respectively. $\mathcal{D}$ represents training data, $x_{1:n}$ represents partially generated response, p represents probability distributions, and x_{n+1} represents the selection of the next token.

the style constraint by replacing θ^{T,S^-} with θ^T that has arbitrary style, and use $\theta^{T,S^\pm}$ to guide style adjustments. The resulting model $\theta^{T,\lambda S}$ will generate responses along the dashed line that passes through the initial point. Second, the initial model can even handle a different task T'. Advancing to Figure 2c, we relax the task constraint by evolving θ^T into $\theta^{T'}$ which operates on a arbitrary task. Although $\theta^{T',\lambda S}$ resides in a different coordinate system, the direction of style adjustment indicated by the dashed line remains consistent with the reference models.

3.2 Baselines

In this subsection, we present three baseline approaches for comparison with our proposed style arithmetic method.

3.2.1 Prompting

The most straightforward approach for linguistic style control is through **Prompting**. We can directly instruct the model to generate responses with specific stylistic characteristics by incorporating style descriptions into the system prompt. This approach requires no additional training and can be applied to any instruction-tuned model. However, it is difficult to achieve fine-grained style variation with this approach, as subtle changes in style are hard to describe explicitly in prompts and the results are heavily influenced by the model's instruction-following capability.

3.2.2 Mixed Supervised Fine-Tuning

Given a pair of training data $\mathcal{D}^{T,S^\pm}$ exhibiting $S^\pm$ styles, another approach for generating responses with style λS is **Mixed Supervised Fine-Tuning** (MSFT). As shown in Figure 3a and 3b, the MSFT process additionally involves creating a mixed training dataset $\mathcal{D}^{T,\lambda S}$ by randomly selecting responses for the style we want to control, where responses with S^- and S^+ styles are selected with probabilities $1-\lambda$ and λ, respectively. Finally, the pretrained model θ_{PT} is fine-tuned on this mixed dataset to produce a model $\theta_{\mathrm{FT}}^{T,\lambda S}$ capable of generating responses exhibiting the desired style λS.

Despite its conceptual simplicity, MSFT has several limitations: it can only handle style intensities between 0 and 1, it cannot transfer styles between different tasks, and it requires separate fine-tuning for each style combination, which is computationally inefficient.

3.2.3 Collaborative Decoding

MSFT necessitates an additional training for each value of λ. To address this issue, we introduce **Collaborative Decoding** (CD), eliminating this overhead by training $\theta_{\text{FT}}^{T,S^{\pm}}$ on their respective datasets $\mathcal{D}^{T,S^{\pm}}$ in advance and fusing them during inference.

At the representation level, **hidden states collaboration** (CD-HS) involves performing a weighted average of the hidden states from the models $\theta^{T,S^{\pm}}$ at each layer, with the weights determined by the respective style intensities. These combined hidden states are then propagated through the network. At the output level, **probability distribution collaboration** (CD-PB), as depicted in Figure 3c, merges the probability distributions from different models using weights based on style intensity.

3.3 Style Arithmetic

While Collaborative Decoding (CD) effectively removes the overhead associated with selecting λ, it necessitates an additional inference for each input instruction, resulting in increased system latency. In contrast, **Style Arithmetic** (SA) circumvents this issue by consolidating $\theta_{\text{FT}}^{T,S^{\pm}}$ into a single model prior to inference.

$$\theta_{\text{SA}}^{T,\lambda S} = (1-\lambda)\,\theta_{\text{FT}}^{T,S^-} + \lambda\theta_{\text{FT}}^{T,S^+} \tag{4}$$

$$= \theta_{\text{FT}}^{T,S^-} + \lambda\left(\theta_{\text{FT}}^{T,S^+} - \theta_{\text{FT}}^{T,S^-}\right), \tag{5}$$

where θ stands for the parameter of a model.

SA offers the capability of style transfer across different tasks. This is because SA operates directly in the parameter space, allowing the style vectors to be applied to any base model, regardless of its original task or style. By relaxing the style and task constraints of the base model $\theta_{\text{FT}}^{T,S^-}$ in Eq. (5), we have:

$$\theta_{\text{SA}}^{T_2,\lambda S} = \theta^{T_2} + \lambda\underbrace{\left(\theta_{\text{FT}}^{T_1,S^+} - \theta_{\text{FT}}^{T_1,S^-}\right)}_{\text{Style vector } \sigma^{T_1,S}}. \tag{6}$$

Drawing inspiration from Ilharco et al. (2023), where task vectors $\tau^T = \theta_{\text{FT}}^T - \theta_{\text{PT}}$ capture task-specific knowledge in parameter space, we introduce the concept of style vectors. A style vector $\sigma^{T,S} = \theta^{T,S^+} - \theta^{T,S^-}$ encapsulates the stylistic characteristics for a given style S in the parameter space. By incorporating style vectors into base model through addition or subtraction, we can strengthen or weaken the presence of the style. Figure 4 illustrates the geometric relationships among these components.

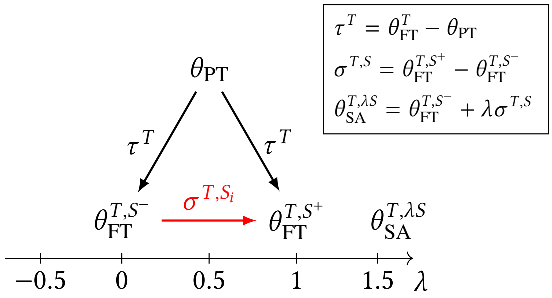


Figure 4: Comparison of style arithmetic and task arithmetic. τ^T is the task vector, and σ^{T,S_i} is the style vector. The points on the axis represent $\theta_{\text{SA}}^{T,\lambda S}$ with different λ.

4 Experiments

4.1 Settings

Task T	**Domain**	**Metric** M^T
Alpaca	Chatting	Response quality
GSM8K	Reasoning	Accuracy
MBPP	Coding	Pass rate

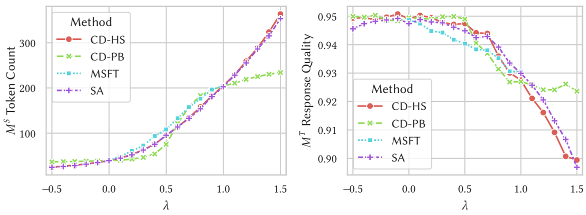
Table 1: Overview of tasks. Response quality is evaluated using Mistral-8B-Instruct (MistralAI, 2024) as a model-based metric, while accuracy and pass rate are rule-based metrics. All the metrics in this table are higher-is-better. See Appendix C.2 for details on using perplexity as an auxiliary metric for response quality, and Appendix B.1 for more information about the tasks.

Dimension S	**Description**	**Metric** M^S
Length (response verbosity)	S^-: Concise S^+: Verbose	Token count
Readability (vocabulary difficulty)	S^-: Easy S^+: Hard	Syllables per word
Complexity (syntactic complexity)	S^-: Simple S^+: Complex	Dependency distance
Sentiment (emotional tone)	S^-: Positive S^+: Negative	Sentiment score
Politeness (social etiquette)	S^-: Polite S^+: Rude	Politeness score

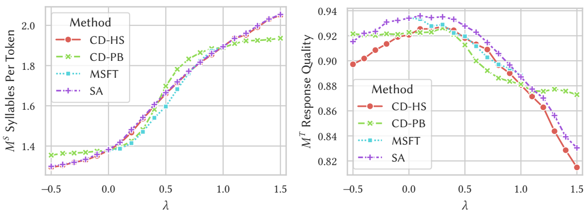
Table 2: Overview of styles. For all metrics, higher values indicate S^+, while lower values indicate S^-. The metrics for sentiment and politeness are model-based classifiers, while the others are based on statistical measures. For more details, please refer to Appendix B.2.

The tasks and styles utilized in the experiments are shown in Table 1 and Table 2, respectively. In Appendix B, we provide a more detailed discussion of our experimental settings. We generate training datasets with diverse styles using GPT-4o-mini (OpenAI et al., 2024). The prompts used for data generation can be found in Appendix D.1. All fine-tuned models in this section are based on Qwen2.5-3B (Qwen et al., 2025). For experiments conducted on the Llama 3 (Grattafiori et al., 2024) family of models, please refer to Appendix C.7.

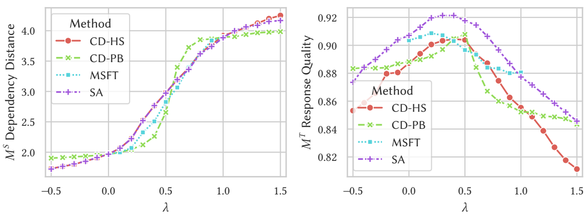
4.2 Study on Controllability



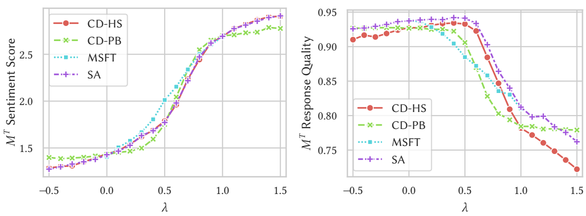
(a) T = Alpaca, S = Length



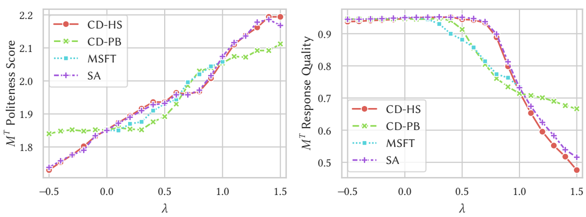
(b) T = Alpaca, S = Readability



(c) T = Alpaca, S = Complexity

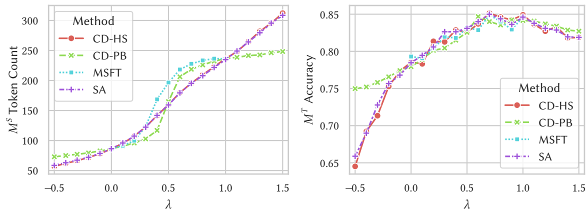


(d) T = Alpaca, S = Sentiment

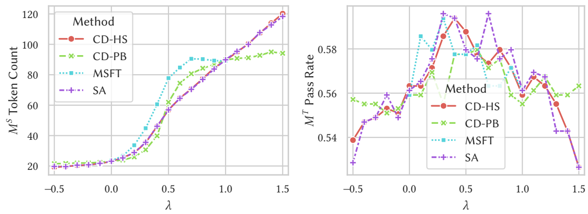


(e) T = Alpaca, S = Politeness

Figure 5: The results of controllability experiments for Alpaca task. Since $\lambda = 0, 1$ correspond to the same fine-tuned models across all methods, both M^S and M^T should be identical at these points theoretically. However, due to different inference libraries used (vLLM for MSFT and SA, Huggingface Transformers for CD), they exhibit minor variations.



(a) T = GSM8K, S = Length



(b) T = MBPP, S = Length

Figure 6: The results of controllability experiments for GSM8K and MBPP tasks. For these domains, only the length style is evaluated, as other linguistic styles are rarely seen. Note that the definition of the length style in GSM8K and MBPP differ from those in Alpaca. Detailed style and task specifications are provided in Appendix D.1.

Settings		**NMSE**$_{\times 10^2}$			
Task	Style	MSFT	CD-PB	CD-HS	SA
Alpaca	Length	0.31	2.78	1.35	1.34
	Read.	0.56	1.31	0.11	0.15
	Comp.	0.63	2.40	0.12	0.13
	Sent.	0.25	2.34	1.55	1.49
	Pol.	0.91	3.24	1.15	1.58
GSM8K	Length	2.52	1.76	0.10	0.10
MBPP	Length	3.75	1.54	0.32	0.31
Average		1.28	2.20	0.67	0.73

Table 3: Normalized mean square error (NMSE) quantifies the deviation between actual and ideal style control curves ($M^S \sim \lambda$). Through normalization, we enable cross-metric comparisons, with lower values reflecting better linearity. See Appendix C.1.1 for detailed methodology.

To validate our core hypothesis that meaningful style representations can be extracted from the parameter space and utilized for precise style control, we first conduct evaluations focusing on individual style-task combinations as formulated in Eq. (2).

For each experiment, we utilize a pair of datasets $\mathcal{D}^{T,S^{\pm}}$ that represent opposing extremes of a style dimension. We construct $R^{T,\lambda S}$ through four distinct methodologies: MSFT, CD-PB, CD-HS, and SA. To evaluate both interpolation and extrapolation capabilities, we vary the style intensity parameter within the range $-0.5 \leq \lambda \leq 1.5$.

The results of our controllability experiments are presented in Figure 5 and Figure 6. Qualitative examples are provided in Appendix E.3. We also explored prompting-based approaches, but due to their reliance on general instruction-following capabilities, these results are presented separately in Appendix C.3.

As shown in Figures 5 and 6, CD-HS and SA methods effectively control style metrics (M^S) while maintaining task performance (M^T). This holds true both for interpolation ($0 < \lambda < 1$) following Eq. (3), and extrapolation ($\lambda < 0$ or $\lambda > 1$) without performance cliffs.

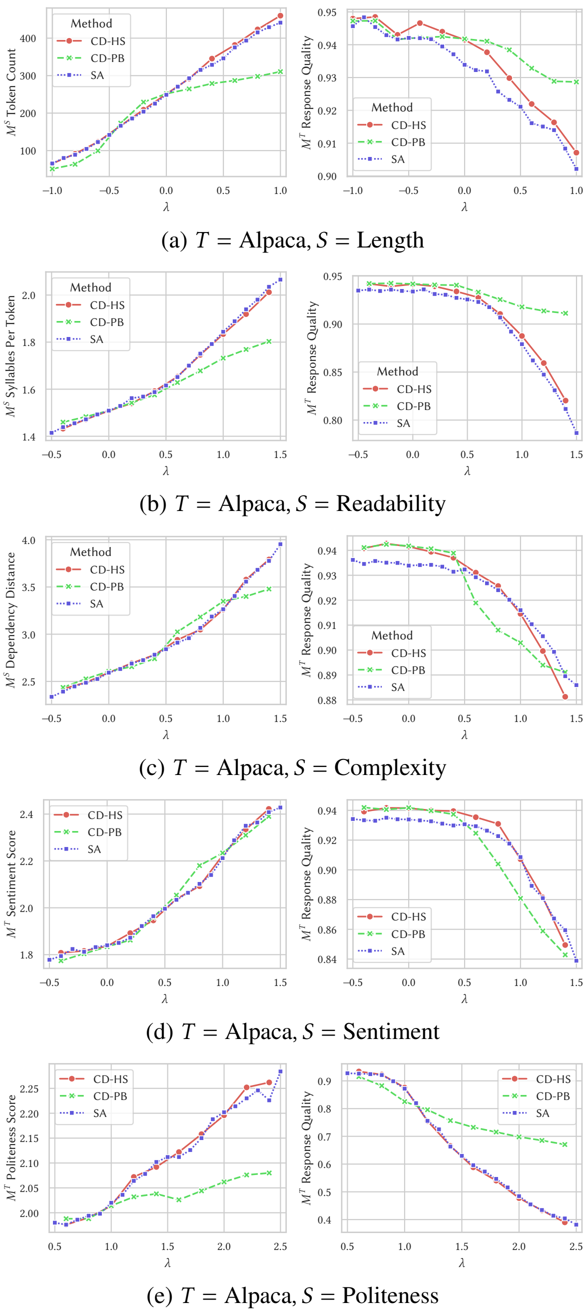
The normalized mean square error presented in Table 3 demonstrates that MSFT and CD-PB methods yield substantially higher error values, suggesting inferior style control capabilities. In particular, CD-PB's poor performance in extrapolation shown in Figure 5 and Figure 6 suggests that interaction between models solely at the final probability distribution level is insufficient. Although CD-HS and SA achieve comparable performance, SA's notably lower computational overhead, as detailed in Appendix A.5, establishes it as the optimal solution for real-world deployment.

For MBPP task, moderate-length responses achieve optimal pass rates, even when generated indirectly through SA or CD. This finding aligns with programming best practices - overly compact code is difficult to write and maintain, while excessively verbose code risks introducing errors.

4.3 Study on Transferability

Settings		Scope Ratio		
Task	Style	CD-PB	CD-HS	SA
Alpaca	Length	1.102	1.240	1.225
	Read.	0.532	0.746	0.762
	Comp.	0.453	0.491	0.510
	Sent.	0.401	0.358	0.351
	Pol.	0.325	0.729	0.693
GSM8K	Length	0.582	0.375	0.377
	Read.	0.347	0.413	0.435
	Comp.	0.420	0.276	0.289
Average		0.520	0.579	0.580

Table 4: The scope ratio measures the relative effectiveness of style transfer by comparing the slopes of M^S curves between transferability and controllability experiments. Due to varying scales across tasks and styles, direct slope comparison is infeasible, necessitating this ratio-based approach. A higher ratio indicates better preservation of style control effectiveness during transfer. See Appendix C.1.2 for calculation details.



(a) T = Alpaca, S = Length

(b) T = Alpaca, S = Readability

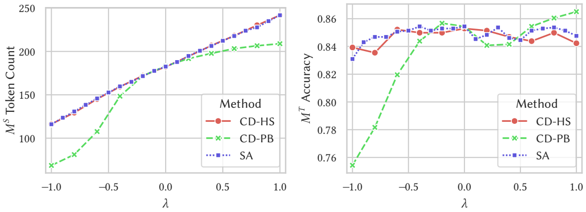
(c) T = Alpaca, S = Complexity

(d) T = Alpaca, S = Sentiment

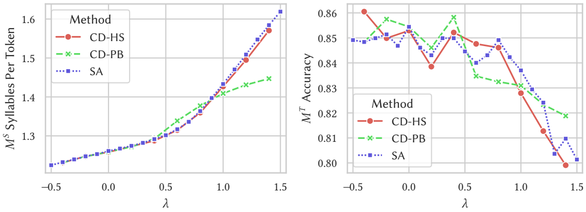
(e) T = Alpaca, S = Politeness

Figure 7: The results of transferability experiments for Alpaca task.

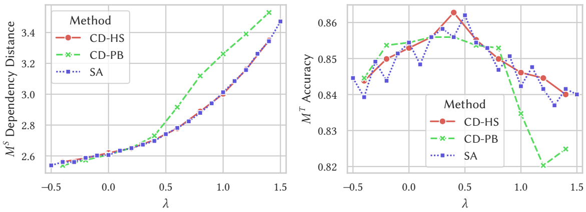
To evaluate style transferability, we leverage the stylized models $\theta^{T,S^{\pm}}$ and datasets $\mathcal{D}^{T,S^{\pm}}$ from the Alpaca task in Section 4.2 to modulate the style of Qwen2.5-3B-Instruct θ_{Ins}, a model enhanced through sophisticated RLHF techniques (Qwen et al., 2025). In our experimental framework, we apply one style dimension at a time and systematically evaluate the transfer effectiveness across two distinct tasks: Alpaca and GSM8K, where Alpaca evaluations assess style transfer within the same task domain (as illustrated in Figure 2b), while



(a) T = GSM8K, S = Length



(b) T = GSM8K, S = Readability



(c) T = GSM8K, S = Complexity

Figure 8: The results of transferability experiments for GSM8K task. We exclude sentiment and politeness styles since their model-based evaluators do not generalize well to mathematical reasoning tasks.

GSM8K evaluations demonstrate cross-task transferability (as shown in Figure 2c).

The experimental results for Alpaca and GSM8K tasks are presented in Figures 7 and 8, respectively. For instance, in Figures 7a and 8a, all data points on the SA curves are generated by the identical model $\theta_{\text{Ins}} + \lambda\sigma^{\text{Alpaca, Length}}$, where $\sigma^{\text{Alpaca, Length}} = \theta_{\text{FT}}^{\text{Alpaca, Verbose}} - \theta_{\text{FT}}^{\text{Alpaca, Concise}}$ represents the style vector for length style derived from the Alpaca task. The only variation between these experiments is the instruction used during evaluation. Additional experiments exploring coding tasks, cross-lingual style transfer, and example data samples are provided in Appendices C.4 and E.3.

Our experimental findings demonstrate successful style transfer across different tasks and methods, with CD-HS and SA achieving comparable control over the style transfer process. This equivalence is evidenced by the similar slope ratios in their respective M^S curves, as quantified in Table 4.

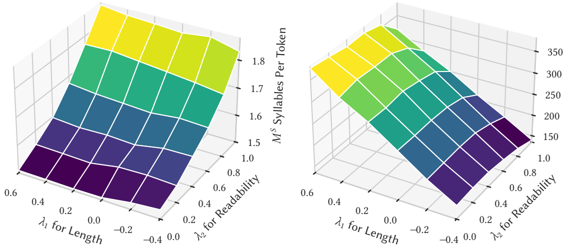
However, our analysis reveals that the effectiveness of style transfer, measured by the slope of the M^S curve, is generally lower in transferability experiments compared to controllability experiments (Section 4.2). This reduction in effectiveness becomes particularly pronounced when transferring styles across different tasks (from Alpaca to GSM8K). We identify two primary factors contributing to this phenomenon: First, there exists an inherent constraint on the range of possible style metric values (M^S) for any valid response. In controllability experiments, we start from an extreme point (S^-) of the style spectrum, allowing for maximum potential change. In contrast, transferability experiments begin from a more neutral starting point, where the initial style may already be closer to the target style (S^+). This naturally results in a smaller scope for style adjustment and consequently a reduced rate of change. Second, the manifestation of linguistic styles varies across different tasks, taking response length as an example - conversational tasks typically expand through additional context and examples, mathematical reasoning requires more detailed step-by-step explanations, and coding tasks benefit from comprehensive comments and modular code structure. These task-specific characteristics create natural boundaries for style transfer, affecting how effectively a style can be adapted across different task domains.

4.4 Study on Composability

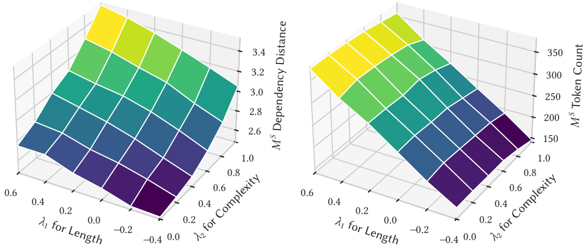
Building on our analysis of single-style transfer presented in Section 4.3, we now extend our investigation to examine the simultaneous application of multiple style dimensions. We focus exclusively on the SA method due to CD method's prohibitive computational requirements — combining just two styles would necessitate loading five models and executing five forward passes, rendering it impractical for real-world applications.

Figure 9 presents our experimental results. As illustrated in Figure 9a, all data points are generated using the model $\theta_{\text{Ins}} + \lambda_1\sigma^{\text{Alpaca, Length}} + \lambda_2\sigma^{\text{Alpaca, Read.}}$, where λ_1 and λ_2 control the intensity of length and readability styles, respectively. The results demonstrate that style metrics M^S exhibit a clear planar relationship with respect to the style intensity parameters λ_1 and λ_2. This consistent pattern across different style combinations provides compelling evidence that the SA method can effectively compose multiple style dimensions simultaneously while maintaining coherent outputs.

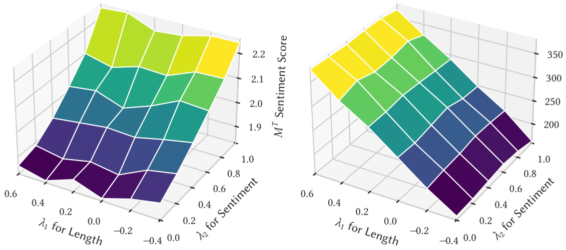
We observe that the styles exhibit interdependence in terms of style metrics, as evidenced in Figure 9b where both λ for length and complexity influence the dependency distance. This interdepen-



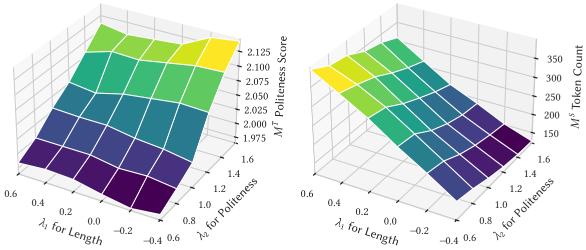
(a) T = Alpaca, S = Length, Readability



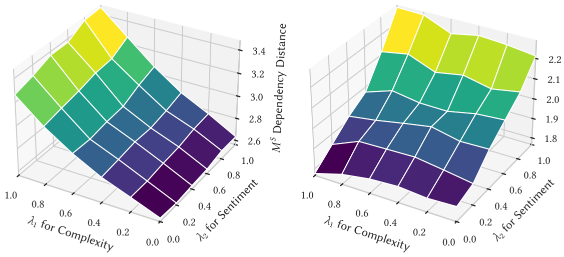
(b) T = Alpaca, S = Length, Complexity



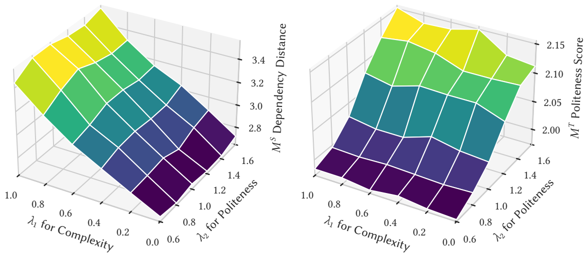
(c) T = Alpaca, S = Length, Sentiment



(d) T = Alpaca, S = Length, Politeness



(e) T = Alpaca, S = Complexity, Sentiment



(f) T = Alpaca, S = Complexity, Politeness

Figure 9: Selected results of composability experiments. For additional style combinations and quality analysis, please refer to Appendix C.6.

dence does not invalidate our previous experimental conclusions regarding the relationship between λ and M^S. We conduct a comprehensive analysis of these interdependencies in Appendix C.5, where we quantify the relationships between different style dimensions. Based on this analysis, the six style combinations presented in Figure 9 were specifically selected because they exhibit the lowest degree of interdependence, allowing us to more clearly demonstrate the effectiveness of our style composition approach while minimizing confounding effects.

5 Conclusion

In this study, we explored the control of linguistic style in language models through three distinct approaches: Mixed Supervised Fine-Tuning (MSFT), Collaborative Decoding (CD), and Style Arithmetic (SA). Our extensive experiments across various tasks and styles revealed that SA emerges as the most effective and practical method, delivering high performance with minimal computational overhead.

6 Limitations

Despite the promising results, several limitations remain. First, developing a rigorous mathematical framework that fully explains the effectiveness of SA represents a significant challenge for future research. Second, both SA and CD-HS approaches are constrained by their reliance on homogeneous model architectures, as they operate directly on model parameters or hidden states. CD-PB necessitates identical vocabularies when merging probability distributions across tokens; nevertheless, heterogeneous language models frequently employ distinct vocabularies, limiting its applicability.

7 Acknowledgement

This work was supported by National Natural Science Foundation of China under Contract 623B2097 and the Youth Innovation Promotion Association CAS. It was also supported by the GPU cluster built by MCC Lab of Information Science and Technology Institution, USTC.

References

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, et al. 2021. Program synthesis with large language models. *Preprint*, arXiv:2108.07732.

Jiangjie Chen, Xintao Wang, Rui Xu, Siyu Yuan, Yikai Zhang, Wei Shi, Jian Xie, Shuang Li, Ruihan Yang, et al. 2024a. From persona to personalization: A survey on role-playing language agents. *Preprint*, arXiv:2404.18231.

Siyuan Chen, Qingyi Si, Chenxu Yang, Yunzhi Liang, Zheng Lin, Huan Liu, and Weiping Wang. 2024b. A multi-task role-playing agent capable of imitating character linguistic styles. *Preprint*, arXiv:2411.02457.

Ziyang Chen and Stylios Moscholios. 2024. Using prompts to guide large language models in imitating a real person's language style. *Preprint*, arXiv:2410.03848.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, et al. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Jasper Dekoninck, Marc Fischer, Luca Beurer-Kellner, and Martin Vechev. 2024. Controlled text generation via language model arithmetic. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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. Association for Computational Linguistics.

Rudolph Flesch. 1948. A new readability yardstick. *Journal of Applied Psychology*.

Tao Ge, Hu Jing, Li Dong, Shaoguang Mao, Yan Xia, Xun Wang, Si-Qing Chen, and Furu Wei. 2023. Extensible prompts for language models on zero-shot language style customization. *Advances in Neural Information Processing Systems*.

Diogo Glória-Silva. 2023. Polite bert. `https://huggingface.co/NOVA-vision-language/polite_bert`.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vlad Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. 2024. Arcee's mergekit: A toolkit for merging large language models. *Preprint*, arXiv:2403.13257.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Robert Gunning. 1969. The fog index after twenty years. *Journal of Business Communication*.

Lasse Hansen, Ludvig Renbo Olsen, and Kenneth Enevoldsen. 2023. Textdescriptives: A python package for calculating a large variety of metrics from text. *Journal of Open Source Software*.

Hieu Hoang, Huda Khayrallah, and Marcin Junczys-Dowmunt. 2024. On-the-fly fusion of large language models and machine translation. *Preprint*, arXiv:2311.08306.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023. Editing models with task arithmetic. In *International Conference on Learning Representations*.

Kai Konen, Sophie Jentzsch, Diaoulé Diallo, Peer Schütt, Oliver Bensch, Roxanne El Baff, Dominik Opitz, and Tobias Hecking. 2024. Style vectors for steering generative large language models. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 782–802, St. Julian's, Malta. Association for Computational Linguistics.

Raymond Li, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, LI Jia, et al. 2023. Starcoder: may the source be with you! *Transactions on Machine Learning Research*.

Tianlin Li, Qian Liu, Tianyu Pang, Chao Du, Qing Guo, Yang Liu, and Min Lin. 2024. Purifying large language models by ensembling a small language model. *Preprint*, arXiv:2402.14845.

Xun Liang, Hanyu Wang, Yezhaohui Wang, Shichao Song, Jiawei Yang, Simin Niu, Jie Hu, Dan Liu, Shunyu Yao, et al. 2024. Controllable text generation for large language models: A survey. *Preprint*, arXiv:2408.12599.

Christopher Liao, Theodoros Tsiligkaridis, and Brian Kulis. 2024. Descriptor and word soups: Overcoming the parameter efficiency accuracy tradeoff for out-of-distribution few-shot learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Haitao Liu. 2008. Dependency distance as a metric of language comprehension difficulty. *Journal of Cognitive Science*.

Xinyue Liu, Harshita Diddee, and Daphne Ippolito. 2024. Customizing large language model generation style using parameter-efficient finetuning. In *Proceedings of the 17th International Natural Language Generation Conference*, pages 412–426, Tokyo, Japan. Association for Computational Linguistics.

G Harry Mc Laughlin. 1969. Smog grading-a new readability formula. *Journal of Reading*.

MistralAI. 2024. Ministral-8b-instruct-2410. https://huggingface.co/mistralai/Ministral-8B-Instruct-2410.

Dang Nguyen, Jiuhai Chen, and Tianyi Zhou. 2024. Multi-objective linguistic control of large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4336–4347, Bangkok, Thailand. Association for Computational Linguistics.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, et al. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

Masanori Oya. 2011. Syntactic dependency distance as sentence complexity measure. In *International Conference of Pan-Pacific Association of Applied Linguistics*.

Vikas Paruchuri. 2023. python-functions-filtered. https://huggingface.co/datasets/vikp/python_functions_filtered.

Nicolas Mejia Petit. 2024. Vezora's codetester dataset. https://huggingface.co/datasets/Vezora/Tested-143k-Python-Alpaca.

Qwen, : An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, et al. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Alexandre Ramé, Guillaume Couairon, Mustafa Shukor, Corentin Dancette, Jean-Baptiste Gaya, Laure Soulier, and Matthieu Cord. 2023. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. In *Annual Conference on Neural Information Processing Systems*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *Preprint*, arXiv:1910.01108.

Ruizhe Shi, Yifang Chen, Yushi Hu, Alisa Liu, Hannaneh Hajishirzi, Noah A Smith, and Simon Shaolei Du. 2024. Decoding-time language model alignment with multiple objectives. In *Annual Conference on Neural Information Processing Systems*.

SilkRoad. 2023. alpaca-data-gpt4-chinese. https://huggingface.co/datasets/silk-road/alpaca-data-gpt4-chinese.

tabularisai. 2023. (distil)bert-based sentiment classification model: Unleashing the power of synthetic data. https://huggingface.co/tabularisai/robust-sentiment-analysis.

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.

Chenning Xu, Fangxun Shu, Dian Jin, Jinghao Wei, and Hao Jiang. 2024. Sag: Style-aligned article generation via model collaboration. *Preprint*, arXiv:2410.03137.

Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. 2024a. Model merging in llms, mllms, and beyond: Methods, theories, applications and opportunities. *arXiv preprint arXiv:2408.07666*.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. 2024b. Adamerging: Adaptive model merging for multi-task learning. In *International Conference on Learning Representations*.

Zhehao Zhang, Ryan A. Rossi, Branislav Kveton, Yijia Shao, Diyi Yang, Hamed Zamani, Franck Dernoncourt, et al. 2024. Personalization of large language models: A survey. *Preprint*, arXiv:2411.00027.

Yuyan Zhou, Liang Song, Bingning Wang, and Weipeng Chen. 2024. Metagpt: Merging large language models using model exclusive task arithmetic. *Preprint*, arXiv:2406.11385.

A Methodology

In this section, we rigorously present our definitions and formalizations, and provide detailed explanations of our methods. We also conduct a thorough analysis of the computational efficiency of different approaches at Appendix A.5.

A.1 Definitions and Formulations

We begin with two fundamental definitions. First, a **task** T is formally defined as a pair $\langle I^T_{\text{Test}}, M^T \rangle$, where I^T_{Test} represents a test instruction set and M^T denotes a performance metric. For instance, mathematical reasoning can be formalized as the task $\langle$GSM8K, Accuracy$\rangle$, where GSM8K provides the test instructions and accuracy serves as the evaluation metric.

Second, we define a **style dimension** as a triple $\langle S^-, S^+, M^S \rangle$, where S^- and S^+ represent two basic contrasting linguistic attributes (e.g., concise versus verbose for the length dimension) that serve as **fundamental styles** from which all other styles can be constructed, and M^S is a quantitative metric that measures the degree of these attributes in text.

The concept of style to encompass the following properties:

- **Closure under affine combination**: For any contrasting styles $S^\pm$ and scalar $\lambda \in \mathbb{R}$, their affine combination yields a valid style:

$$\lambda S = (1-\lambda)\, S^- + \lambda S^+. \tag{7}$$

  where λ is referred to as the **style intensity**. Here, $S^\pm$ can be regarded as specific instances of the style λS when λ is either 0 or 1. For $0 < \lambda < 1$, λS represents an interpolation between S^- and S^+; otherwise, it signifies an extrapolation of S^- and S^+.

- **Closure under addition**: For any collection of styles $\{\lambda_i S_i\}_{i=1}^m$, their summation yields a valid style:

$$\sum_{i=1}^m \lambda_i S_i = \sum_{i=1}^m \left[(1-\lambda_i)\, S_i^- + \lambda_i S_i^+ \right]. \tag{8}$$

A.1.1 Controllability

Our linguistic control framework consists of several key components built upon a pretrained model θ_{PT}, a task $T = \langle I^T_{\text{Test}}, M^T \rangle$ associated with training instructions I^T_{Train}, and a style dimension $S = \langle S^-, S^+, M^S \rangle$. We first generate stylized responses $R^{T,S^\pm}_{\text{Train}}$ for the training instruction set I^T_{Train}, representing the desired manifestation of style $S^\pm$ on task T. Through this process, we construct training sets $\mathcal{D}^{T,S^\pm} = \langle I^T_{\text{Train}}, R^{T,S^\pm}_{\text{Train}} \rangle$. We then create models $\theta^{T,S^\pm}_{\text{FT}}$ through supervised fine-tuning θ_{PT} on $\mathcal{D}^{T,S^\pm}$. These models ultimately generate stylized test responses $R^{T,S^\pm}_{\text{Test}}$ when given test instructions I^T_{Test}.

Our primary objective is to develop an efficient method for generating responses $R^{T,\lambda S}_{\text{Test}}$ that align with the style λS, utilizing the training datasets $\mathcal{D}^{T,S^\pm}$ and the models $\theta^{T,S^\pm}_{\text{FT}}$. The expected $M^S\left(R^{T,\lambda S}_{\text{Test}}\right)$ is specified by $M^S\left(R^{T,S^\pm}_{\text{Test}}\right)$, which can be expressed as:

$$\underbrace{M^S\left(R^{T,\lambda S}_{\text{Test}}\right)}_{\text{Metric of }\lambda S} = M^S\left(R^{T,(1-\lambda)S^- + \lambda S^+}_{\text{Test}}\right) \tag{9}$$

$$\approx (1-\lambda)\underbrace{M^S\left(R^{T,S^-}_{\text{Test}}\right)}_{\text{Metric of }S^-} + \lambda \underbrace{M^S\left(R^{T,S^+}_{\text{Test}}\right)}_{\text{Metric of }S^+} \tag{10}$$

$$= \underbrace{M^S\left(R^{T,S^-}_{\text{Test}}\right)}_{\text{Initial point}} + \lambda \underbrace{\left[M^S\left(R^{T,S^+}_{\text{Test}}\right) - M^S\left(R^{T,S^-}_{\text{Test}}\right)\right]}_{\text{Controlling slope}}. \tag{11}$$

The approximation in Eq. (10) illustrates our expectation to linearly control the linguistic style of responses $R^{T,\lambda S}_{\text{Test}}$ using λ. This relationship is guaranteed to hold when λ is either 0 or 1, regardless of the employed method, since $S^- = \lambda S|_{\lambda=0}$ and $S^+ = \lambda S|_{\lambda=1}$ by definition. It is important to emphasize that λ is not constrained to the interval $[0,1]$ in our formulation, allowing for a broader range of style control.

For a composed style that combines multiple individual styles $\sum_{j=1}^m \lambda_j S_j$, our goal is to generate $R^{T,\sum_{j=1}^m \lambda_j S_j}_{\text{Test}}$ that satisfies:

$$\underbrace{M^{S_i}\left(R^{T,\sum_{j=1}^m \lambda_j S_j}_{\text{Test}}\right)}_{\text{Style of }\sum_{j=1}^m \lambda_j S_j} \approx \sum_{j=1}^m \underbrace{M^{S_i}\left(R^{T,\lambda_j S_j}_{\text{Test}}\right)}_{\text{Style of }\lambda_j S_j} \tag{12}$$

$$\approx \sum_{j=1}^m \left[(1-\lambda_j)\underbrace{M^{S_i}\left(R^{T,S_j^-}_{\text{Test}}\right)}_{\text{Style of }S_j^-} + \lambda_j \underbrace{M^{S_i}\left(R^{T,S_j^+}_{\text{Test}}\right)}_{\text{Style of }S_j^+}\right] \tag{13}$$

$$= \underbrace{\sum_{j=1}^m M^{S_i}\left(R^{T,S_j^-}_{\text{Test}}\right)}_{\text{Initial point}} + \sum_{j=1}^m \lambda_j \underbrace{\left[M^{S_i}\left(R^{T,S_j^+}_{\text{Test}}\right) - M^{S_i}\left(R^{T,S_j^-}_{\text{Test}}\right)\right]}_{\text{Controlling slope for }S_j} \tag{14}$$

for $i = 1, ..., m$. The expected metrics for $\sum_{j=1}^m \lambda_j S_j$ are described by $M^{S_i}\left(S_j^\pm\right), i, j = 1, ..., m$, through a two-step decomposition, namely Eq. (12) and (13), corresponding to the two properties of style, which are Eq. (8) and (7), respectively. It is important to note that the approximation in Eq. (12) does not guarantee validity, indicating that the initial point in Eq. (14) is method dependent.

A direct implication of Eq. (10) is that, for the style metric M^S, we have:

$$M^S\left(R^{T,\lambda S}\right) \approx \frac{\lambda-\lambda_1}{\lambda_2-\lambda_1} M^S\left(R^{T,\lambda_1 S}\right)+\frac{\lambda_2-\lambda}{\lambda_2-\lambda_1} M^S\left(R^{T,\lambda_2 S}\right). \tag{15}$$

Assuming a local linear relationship between M^S and M^T, which implies that small changes in style intensity incur a constant quality cost, our requirements for response quality can be formalized as:

$$M^T\left(R^{T,\lambda S}\right) \gtrsim \frac{\lambda-\lambda_1}{\lambda_2-\lambda_1} M^T\left(R^{T,\lambda_1 S}\right)+\frac{\lambda_2-\lambda}{\lambda_2-\lambda_1} M^T\left(R^{T,\lambda_2 S}\right). \tag{16}$$

When $0 \le \lambda \le 1$, we set $\lambda_1 = 0$ and $\lambda_2 = 1$, resulting in

$$M^T\left(R^{T,\lambda S}\right) \gtrsim (1-\lambda) M^T\left(R^{T,S^-}\right)+\lambda M^T\left(R^{T,S^+}\right). \tag{17}$$

In interpolation experiments, we further strengthen the relation $\gtrsim$ to $\ge$, as shown in Eq. (3). For $\lambda > 1.0$ (the situation is similar for $\lambda < 0$), taking $\lambda > \lambda_2 > \lambda_1$, Eq. (16) reflects that $M^T\left(R^{T,\lambda S}\right)$ should roughly lie above the line connecting points $\left(\lambda_1, M^T\left(R^{T,\lambda_1 S}\right)\right)$ and $\left(\lambda_2, M^T\left(R^{T,\lambda_2 S}\right)\right)$. In other words, the M^T curve should not exhibit cliff-like degradation.

A.1.2 Transferability

Equation (11) offers a novel perspective on style control. It begins with an initial point with style S^- on task T, and leverages $\mathcal{D}^{T,S^\pm}$ along with $\theta_{\text{FT}}^{T,S^\pm}$ for the same task to adjust its linguistic style according to style intensity λ. It is important to note that the conditions for the initial point can be relaxed. First, its style is not confined to S^-; any model with an arbitrary style can serve as the initial point. Second, its task is not restricted to T; many tasks often exhibit similarities in linguistic styles, which enhances the transferability of these styles across different tasks.

To formulate style transfer, we establish a framework consisting of two series of components. For the initial point, we employ a model $\theta^{T'}$ that generates responses $R_{\text{Test}}^{T'}$ for task T' based on test instructions $I_{\text{Test}}^{T'}$, with an arbitrary initial style. In order to control the style S of the initial point, we utilize the styled training dataset $\mathcal{D}^{T,S^\pm}$ and fine-tuned models $\theta_{\text{FT}}^{T,S^\pm}$ from another task T. Our goal is to generate responses $R_{\text{Test}}^{T',\lambda S}$ for task T' that inherit the style characteristics $S^\pm$ manifesting on task T while meeting specific metric requirements:

$$M^S\left(R_{\text{Test}}^{T',\lambda S}\right) \approx \underbrace{M^S\left(R_{\text{Test}}^{T'}\right)}_{\substack{\text{Initial point with}\\ \text{arbitrary style on } T'}} + \lambda c \underbrace{\left[M^S\left(R_{\text{Test}}^{T,S^+}\right)-M^S\left(R_{\text{Test}}^{T,S^-}\right)\right]}_{\substack{\text{Controlling scope specified by}\\ \text{a different task } T}} \tag{18}$$

where c is a constant scalar, accounting for the control effectiveness degradation during the transfer process.

For a composed style represented as $\sum_{j=1}^m \lambda_j S_j$, our goal is to generate responses $R_{\text{Test}}^{T',\sum_{j=1}^m \lambda_j S_j}$ that satisfy:

$$M^{S_i}\left(R_{\text{Test}}^{T',\sum_{j=1}^m \lambda_j S_j}\right) \approx \underbrace{M^{S_i}\left(R_{\text{Test}}^{T'}\right)}_{\substack{\text{Initial point with}\\ \text{arbitrary style on } T'}} + \sum_{j=1}^m \lambda_j c_j \underbrace{\left[M^{S_i}\left(R_{\text{Test}}^{T_j,S_j^+}\right)-M^{S_i}\left(R_{\text{Test}}^{T_j,S_j^-}\right)\right]}_{\substack{\text{Controlling scope specified by}\\ \text{different tasks } T_j}} \tag{19}$$

In contrast to Eq. (14), the initial point is method independent. The approximation in Eq. (19) holds true when $\lambda_j = 0$ for all j.

A.2 Mixed Supervised Fine-Tuning

MSFT operates by fine-tuning θ_{PT} on a mixed dataset constructed from $\mathcal{D}^{T,S_i^\pm}$.

To begin, individual training sets $\mathcal{D}^{T,\lambda_i S_i} = \langle I_{\text{Train}}^T, R_{\text{Train}}^{T,\lambda_i S_i}\rangle$ are constructed for each linguistic style $\lambda_i S_i$. For each instruction in I_{Train}^T, responses are randomly selected from either $R_{\text{Train}}^{T,S_i^-}$ or $R_{\text{Train}}^{T,S_i^+}$, with probabilities of $(1-\lambda_i)$ and λ_i, respectively.

Next, a proportion μ_i of the data from each training set $\mathcal{D}^{T,\lambda_i S_i}$ is selected to form the mixed training set $\mathcal{D}^{T,\sum_{i=1}^m \lambda_i S_i}$. If the condition $\sum_{i=1}^m \mu_i = 1$ holds, then it follows that $|\mathcal{D}^{T,\sum_{i=1}^m \lambda_i S_i}| = |I_{\text{Train}}^T|$.

Subsequently, the pretrained model θ_{PT} is fine-tuned on the mixed dataset $\mathcal{D}^{T,\sum_{i=1}^m \lambda_i S_i}$, resulting in the model $\theta_{\text{FT}}^{T,\sum_{i=1}^m \lambda_i S_i}$. The responses $R_{\text{Test}}^{T,\sum_{i=1}^m \lambda_i S_i}$ can then be generated by inferencing with the fine-tuned model.

A.3 Collaborative Decoding

We can also establish collaboration between multiple models during or after the inference process.

A.3.1 Hidden States

The first approach is to aggregate the hidden states from multiple models. For each layer, we compute a weighted average of hidden states across all models, which then serves as input to the subsequent layer of all models. This process can be formalized mathematically as follows:

$$h_{\text{CD}}^{l} = \sum_{i=1}^{m} \mu_i \left[(1-\lambda_i)\, h^{l}_{\theta_{\text{FT}}^{T,S_i^-}} + \lambda_i h^{l}_{\theta_{\text{FT}}^{T,S_i^+}} \right] \quad (20)$$

$$h^{l+1}_{\theta_{\text{FT}}^{T,S_i^\pm}} = L^{l+1}_{\theta_{\text{FT}}^{T,S_i^\pm}} \left(h_{\text{CD}}^{l} \right) \quad (21)$$

Here, L_θ^l is the l-th layer of model θ, and h_θ^l represents the hidden state of model θ at layer l. μ_i is the weight assigned to the style dimension S_i, constrained by $\sum_{i=1}^{m} \mu_i = 1$.

A.3.2 Probability

The second approach is computing a weighted average of the probability from multiple models for further sampling:

$$\hat{p}_{\text{CD}} = \sum_{i=1}^{m} \mu_i \left[(1-\lambda_i)\, p_{\theta_{\text{FT}}^{T,S_i^-}} + \lambda_i p_{\theta_{\text{FT}}^{T,S_i^+}} \right] \quad (22)$$

$$p_{\text{CD}} = \frac{\text{Clip}\left(\hat{p}_{\text{CD}}\right)}{\sum_{i=1}^{m} \text{Clip}\left(\hat{p}_{\text{CD}}\right)} \quad (23)$$

Here, p_θ represents the probability distribution over the next token produced by model θ. To ensure the final output is a valid probability distribution, we clip the values and normalize them.

CD is capable of transferring styles across different tasks. Taking probability p as an example:

$$\hat{p}_{\text{CD}} = (1-\lambda)\, p_{\theta_{\text{FT}}^{T,S^-}} + \lambda p_{\theta_{\text{FT}}^{T,S^+}} \quad (24)$$

$$= p_{\theta_{\text{FT}}^{T,S^-}} + \lambda \left(p_{\theta_{\text{FT}}^{T,S^+}} - p_{\theta_{\text{FT}}^{T,S^-}} \right) \quad (25)$$

By relaxing the style and task constraints of the initial point $p_{\theta_{\text{FT}}^{T,S^-}}$ and extending it to multiple styles, we obtain:

$$\hat{p}_{\text{CD}}^{T',\sum_{i=1}^{m}\lambda_i S_i} = p_{\theta^{T'}} + \sum_{i=1}^{m} \lambda_i \left(p_{\theta_{\text{FT}}^{T_i,S_i^+}} - p_{\theta_{\text{FT}}^{T_i,S_i^-}} \right) \quad (26)$$

where $p_{\theta^{T'}}$ is the probability of the model $\theta^{T'}$, with arbitrary style and task T', and $p_{\theta_{\text{FT}}^{T_i,S_i^\pm}}$ are the probabilities of the two fine-tuned models $\theta_{\text{FT}}^{T_i,S_i^\pm}$, respectively.

A.4 Style Arithmetic

SA merges $\theta_{\text{FT}}^{T,S_i^\pm}$ into a single model before inference.

$$\theta_{\text{SA}}^{T,\sum_{i=1}^{m}\lambda_i S_i} = \sum_{i=1}^{m} \mu_i \left[(1-\lambda_i)\, \theta_{\text{FT}}^{T,S_i^-} + \lambda_i \theta_{\text{FT}}^{T,S_i^+} \right] \quad (27)$$

$$= \underbrace{\sum_{i=1}^{m} \mu_i \theta_{\text{FT}}^{T,S_i^-}}_{\text{Base model}} + \sum_{i=1}^{m} \mu_i \lambda_i \underbrace{\left(\theta_{\text{FT}}^{T,S_i^+} - \theta_{\text{FT}}^{T,S_i^-} \right)}_{\text{Style vector } \sigma^{T,S_i}} \quad (28)$$

where θ stands for the parameter of a model. Then $R_{\text{Test}}^{T,\sum_{i=1}^{m}\lambda_i S_i}$ could be generated by decoding with $\theta_{\text{SA}}^{T,\sum_{i=1}^{m}\lambda_i S_i}$. It is worth noting that the term μ_i in the style vector part of Eq. (28) can be omitted.

For transferability, we can first consider the base model in Eq. (28) as a whole, and then relax its style and task constraints. By removing these constraints, we have:

$$\theta_{\text{SA}}^{T',\sum_{i=1}^{m}\lambda_i S_i} = \theta^{T'} + \sum_{i=1}^{m} \lambda_i \underbrace{\left(\theta_{\text{FT}}^{T_i,S_i^+} - \theta_{\text{FT}}^{T_i,S_i^-} \right)}_{\text{Style vector } \sigma^{T_i,S_i}} \quad (29)$$

A.5 Efficiency

Number of	Plain Control			Transfer	
	MSFT	CD	SA	CD	SA
Initial preparation for m styles and n tasks					
Datasets	$2mn$	$2mn$	$2mn$	$2m$	$2m$
Fine-tuning	0	$2mn$	$2mn$	$2m$	$2m$
System construction for each $\lambda_1, ..., \lambda_m$ selection					
Fine-tuning*	1	0	0	0	0
Multiplication	0	0	m	0	m
Inference for each instruction					
Inferences*	1	$2m$	1	$2m+1$	1

Table 5: A summary of the efficiency of different methods. Minor computational costs such as data mixing and probability/hidden state fusion are omitted. Operations marked with * indicate substantial computational overhead and should be avoided whenever possible.

Our system allows users to define a parameter λ, which indicates the desired intensity of the response style. Based on this parameter and the user's input instruction, the system generates tailored responses that reflect the specified style. To evaluate the practical implications of implementing such a system, we analyze the computational efficiency of all the methods across three key operational phases: initial preparation, system construction, and inference. The initial preparation phase involves one-time costs that apply universally across all λ selections and user instructions. The system construction phase occurs whenever a user specifies a new λ value, requiring system reconfiguration. Finally, the inference phase represents the computational cost for each user instruction.

While all approaches require paired training datasets ($\mathcal{D}^{T,S^{\pm}}$) representing style endpoints $S^{\pm}$, they differ significantly in operational characteristics. MSFT, despite having no initial preparation cost, requires dataset mixing and model fine-tuning during system construction, making it computationally intensive when adapting to new λ values. SA necessitates two initial model fine-tunings to obtain $\theta^{T,S^{\pm}}$ and style vector computation $\sigma^{T,S} = \theta^{T,S^{+}} - \theta^{T,S^{-}}$ (one addition per parameter), but its system construction phase only involves simple parameter arithmetic $\theta_{\text{SA}} = \theta + \lambda\sigma^{T,S}$ (one addition and one multiplication per parameter). CD also necessitates two initial model fine-tunings; however, it circumvents system construction costs, albeit at the expense of doubling the inference overhead due to the requirement of two forward passes along with state or probability mixing.

Given that initial preparation costs are amortized over time and system construction costs can be distributed across multiple inferences, SA emerges as the most efficient approach. MSFT's requirement for model fine-tuning at each λ selection is typically prohibitive, while CD's doubled inference latency makes it less efficient than SA. When the number of styles increases, as shown in Table 5, the required number of inferences for CD grows proportionally, making the efficiency gap more pronounced.

We also analyze the efficiency gains from style transfer, which is applicable to SA and CD, in Table 5. With style transfer, we can apply all styles learned from one task to other tasks, reducing the required datasets and models in the preparation phase from $2mn$ to $2m$. For m styles, SA needs to add m style vectors to the base model during system construction whether transfer or not. For the CD method, the need for $2m + 1$ inferences during style transfer is explained by Equation (26), where we need two inferences for each style dimension plus one for the base model.

B Experiment Settings

In this section, we present our experimental settings from three perspectives: tasks, styles, and models. We systematically examine how these components interact to evaluate the effectiveness of our approach to linguistic style control.

B.1 Tasks

We conducted experiments across three primary application areas of language models: chatting, mathematical reasoning, and code writing. Each area features distinct tasks defined by specific test sets and evaluation metrics, with some datasets having corresponding training sets for finetuning pretrained models.

B.1.1 Chatting

Alpaca: The original Stanford Alpaca dataset (Taori et al., 2023) comprises 52,000 instruction-input-response pairs focused on everyday life scenarios. We first filtered this dataset to exclude any data related to code and mathematics using GPT-4o-mini with the prompts in Appendix D.2. After filtering, we sampled 500 instruction-input pairs for testing and 7,000 pairs for creating a training instruction set aimed at generating data with specific linguistic styles. To maintain consistency with other datasets, we concatenate the original instructions and inputs together as new instructions.

To evaluate response quality, we employed Ministral-8B-Instruct (MistralAI, 2024) as an evaluator to assess whether responses adequately addressed the given instructions. The evaluation process involves prompting the evaluator to classify each response as either "proper" or "improper" based on specific criteria (see Appendix D.3 for detailed prompt). In our evaluation framework, we aimed to focus solely on response content by explicitly instructing the evaluator to disregard stylistic elements such as response length, vocabulary difficulty, and sentence structure complexity. However, we acknowledge that some inherent biases may persist despite these precautions. The evaluation criteria also specifically identify abnormal language patterns, such as repetitive content or meaningless text, as grounds for an "improper" classification.

Based on the evaluator's output probabilities for "proper" and "improper" classifications, we calculate a quality score for each response using the following formula:

$$\text{Response Quality} = \frac{1}{N}\sum_{r\in R}\frac{p^{+}(r)}{p^{+}(r)+p^{-}(r)} \quad (30)$$

where r represents a response in R, $p^{+}(r)$ and $p^{-}(r)$ represent the probabilities of the evaluator generating "proper" and "improper" for response r, respectively, and N is the total number of responses in R. We refer to this as the **response quality**, which ranges from 0 to 1, with higher values indicating better performance.

Additionally, we employ perplexity (PPL) as a supplementary metric to evaluate response quality in Appendix C.2.

Alpaca Chinese: This task serves as the Chinese counterpart of Alpaca task, primarily used to validate the transferability of linguistic styles across languages. We sampled 500 instructions from SilkRoad (2023) as testing instructions and evaluated the responses using the same methodology as the Alpaca task.

B.1.2 Mathematical Reasoning

GSM8K: GSM8K (Cobbe et al., 2021) consists of 8.5K high quality grade school math problems created by human problem writers, with problems requiring 2-8 steps to solve using basic arithmetic operations. We utilized the dataset's original split of 7500 training problems and 1319 test problems, regenerating the step-by-step problem-solving processes in accordance with concise and verbose styles. **Accuracy** serves as the evaluation metric, focusing on the consistency of numerical answers. The accuracy is calculated as the proportion of correct answers among all test samples:

$$\text{Accuracy} = \frac{1}{N} \sum_{r \in R} \mathbf{1}_c(r) \tag{31}$$

where $\mathbf{1}_c$ is an indicator function that equals 1 if the response r is correct and 0 otherwise. To extract answers from the model's responses, we employ three methods sequentially: first, by locating "The answer is: " and extracting the subsequent number; if that fails, we search for `\boxed{}` to extract the number within; if neither method yields results, we select the last number from the entire response as the answer.

B.1.3 Coding

MBPP: The MBPP dataset (Austin et al., 2021) consists of Python programming problems designed for entry-level programmers, covering fundamental programming concepts and standard library functionality. Each problem provides a function signature and docstring, requiring the model to complete the function implementation. Due to limited training data in original MBPP dataset, we additionally sampled 7000 training examples from Paruchuri (2023) which is extracted and filtered from the starcoder (Li et al., 2023) training data, and used GPT-4o-mini to ensure all these training examples adhered to the same data format. We evaluated model performance on the original test set by executing each response against the provided test cases. The model's performance is quantified using the **pass rate** metric:

$$\text{Pass rate} = \frac{1}{N} \sum_{r \in R} \mathbf{1}_s(r) \tag{32}$$

where $\mathbf{1}_s$ is an indicator function that equals 1 if the response r passes all test cases and 0 otherwise. This metric reflects the proportion of responses that the model's implementation successfully passes.

Vezora: Vezora refers to the CodeTester Dataset (Petit, 2024), which requires models to first analyze problems using natural language and then provide complete implementations in code blocks. We use the **pass rate** as shown in (32) as the evaluation metric. During evaluation, we extract all code blocks from the responses, considering it as pass if all code blocks execute successfully.

B.2 Styles

In our experiments, we investigated five style dimensions, each consisting of a pair of opposite styles and a corresponding evaluation metric. Among them, length, readability, and complexity are measured using statistical metrics, while sentiment and politeness are evaluated using model-based classifiers.

B.2.1 Length

This style indicates the brevity and verbosity of the response, assessed using **token count** calculated by `TextDescriptives` (Hansen et al., 2023). A lower token count typically reflects a more concise response, while a higher count suggests a more verbose one.

B.2.2 Readability

This style measures the difficulty of the words in the response. We use the **syllable per word** calculated by `TextDescriptives` to check how well readers can understand the text. The idea behind this way of measuring is that easier-to-read text usually uses shorter words, which have fewer syllables. For example, in the sentence "The book is easy to read", the word "easy" has one syllable, while in "The manuscript is comprehensible", the word "comprehensible" has four syllables. This illustrates how word choices with different syllable counts can affect the readability of text, even when expressing similar meanings. It is worth noting that our definition of readability differs from conventional readability metrics like Gunning-Fog (Gunning, 1969), SMOG (Mc Laughlin, 1969), and Flesch reading ease (Flesch, 1948). While traditional metrics also consider sentence-level complexity and word-level difficulty, we deliberately focus only on word-level difficulty to avoid redundancy with other metrics.

For code data, since pure code does not exhibit this linguistic style, we only retain the comments during the evaluation, though these comments often contain identifiers and symbols related to the code. For mathematical data, we define rules to remove displayed equations, while keeping inline equations due to their diverse forms and frequent appearance as part of sentences. It is important to note that numbers and symbols are typically tokenized separately, so the average syllable count of code and mathematical data cannot be directly compared to that of plain text. For Chinese data, we do not evaluate readability since Chinese characters do not have the concept of syllables. The syllable-based readability metric is not applicable to Alpaca Chinese.

B.2.3 Complexity

This style reflects the intricacy of sentence structures in the response, evaluated through **dependency distance** calculated by `TextDescriptives`. Dependency distance is a metric that quantifies the average distance between words in a sentence based on their grammatical relationships. A higher dependency distance often indicates more complex sentence structures (Oya, 2011; Liu, 2008), as it suggests that words are more spread out and may involve more intricate syntactic connections, while a lower distance indicates simpler sentence structures. For example, in the sentence "The cat that chased the mouse ran away", the dependency distance between "cat" and "ran" is 5 (counting the words in between), while in "The cat ran away after chasing the mouse", the dependency distance between "cat" and "ran" is only 1. This illustrates how different sentence structures can lead to varying dependency distances.

For the same reasons as readability, we only evaluate the comments extracted from the code data and non-displayed equations from the mathematical data.

B.2.4 Sentiment

This style dimension captures the emotional polarity expressed in a response, ranging from positive to negative. For automatic evaluation, we employ the tabularisai/robust-sentiment-analysis (tabularisai, 2023) model, a fine-tuned version of distilbert/distilbert-base-uncased (Sanh et al., 2020), to classify sentiment. The classifier assigns each response to one of five categories: Very Negative, Negative, Neutral, Positive, or Very Positive. To quantify this dimension, we introduce the **sentiment score**, which is mapped from the classifier's categorical output to an integer value.

To ensure consistency with other style metrics—where a higher value indicates a stronger tendency toward S^+ — we map the sentiment categories to integer scores as follows:

$$\text{Sentiment score} = \begin{cases} 4, & \text{if } r \text{ is Very Negative} \\ 3, & \text{if } r \text{ is Negative} \\ 2, & \text{if } r \text{ is Neutral} \\ 1, & \text{if } r \text{ is Positive} \\ 0, & \text{if } r \text{ is Very Positive} \end{cases} . \tag{33}$$

In practice, we found that this sentiment classifier does not generalize well to mathematical and coding domains. Therefore, we do not conduct sentiment style experiments in these domains.

B.2.5 Politeness

This style dimension quantifies the degree of politeness in a response, ranging from polite to impolite. We use the NOVA-vision-language/polite_bert classifier (Glória-Silva, 2023) (based on google-bert/bert-base-uncased (Devlin et al., 2019)) to automatically evaluate politeness. The classifier outputs four categories: Not Polite, Neutral, Somewhat Polite, and Polite. We define the **politeness score** as an integer mapped from these categories, where higher scores indicate greater impoliteness:

$$\text{Politeness score} = \begin{cases} 3, & \text{if } r \text{ is Not Polite} \\ 2, & \text{if } r \text{ is Neutral} \\ 1, & \text{if } r \text{ is Somewhat Polite} \\ 0, & \text{if } r \text{ is Polite} \end{cases} \tag{34}$$

Consistent with our treatment of the sentiment dimension, we do not assess politeness in code or mathematical domains.

B.3 Models

To prepare paired training datasets with contrasting styles, we use GPT-4o-mini (OpenAI et al., 2024) to generate responses with distinct styles. For the Alpaca dataset, we created 10 distinct training datasets by generating response pairs across 5 style dimensions: response length (Alpaca Concise/Verbose), readability (Alpaca Easy/Hard), complexity (Alpaca Simple/Complex), sentiment (Alpaca Negative/Positive), and politeness (Alpaca Impolite/Polite). For GSM8K and MBPP tasks, we focus only on response length, resulting in GSM8K Concise/Verbose and Starcoder Concise/Verbose datasets. The Vezora dataset was used in its original form without style variations, while the Alpaca-Chinese dataset served solely as a test set for evaluating cross-lingual style transfer capabilities.

We trained dedicated language models on each of these 15 training datasets (Alpaca 10, GSM8K 2, MBPP 2, Vezora 1) to support CD and SA experiments. We also incorporated instruction-tuned variants of our pre-trained models, specifically Qwen2.5-3B-Instruct (Qwen et al., 2025), which were instrumental in evaluating style transferability and composability.

C More Experiments

This section provides supplementary materials and additional experimental results for Section 4.

C.1 Explanation of Normalized Mean Square Error and Slope Ratio

In this section, we explain the metrics used in Table 3 and Table 4.

C.1.1 Normalized Mean Square Error

Normalized Mean Square Error (NMSE) is used to quantitatively evaluate the error between the ideal M^S curve and the actual M^S curve, and can be compared across tasks and styles. The calculation is divided into two steps. First, normalize the actual M^S curve by mapping $M^S\left(R^{T,S^{\pm}}\right)$ to 0 and 1 respectively. The normalized actual M^S curve could be represented as

$$\hat{M}^S\left(R^{T,\lambda S}\right) = \frac{M^S\left(R^{T,\lambda S}\right) - M^S\left(R^{T,S^-}\right)}{M^S\left(R^{T,S^+}\right) - M^S\left(R^{T,S^-}\right)} \tag{35}$$

Correspondingly, the normalized ideal M^S curve is a straight line passing through $(0,0)$ and $(1,1)$.

$$\tilde{M}^S\left(R^{T,\lambda S}\right) = \lambda \tag{36}$$

Then, calculate the NMSE as the mean square error between the normalized actual M^S curve and the ideal M^S curve.

$$\text{NMSE} = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{M}^S\left(R^{T,\lambda_i S}\right) - \tilde{M}^S\left(R^{T,\lambda_i S}\right)\right)^2 \tag{37}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\left(\hat{M}^S\left(R^{T,\lambda_i S}\right) - \lambda_i\right)^2 \tag{38}$$

C.1.2 Slope Ratio

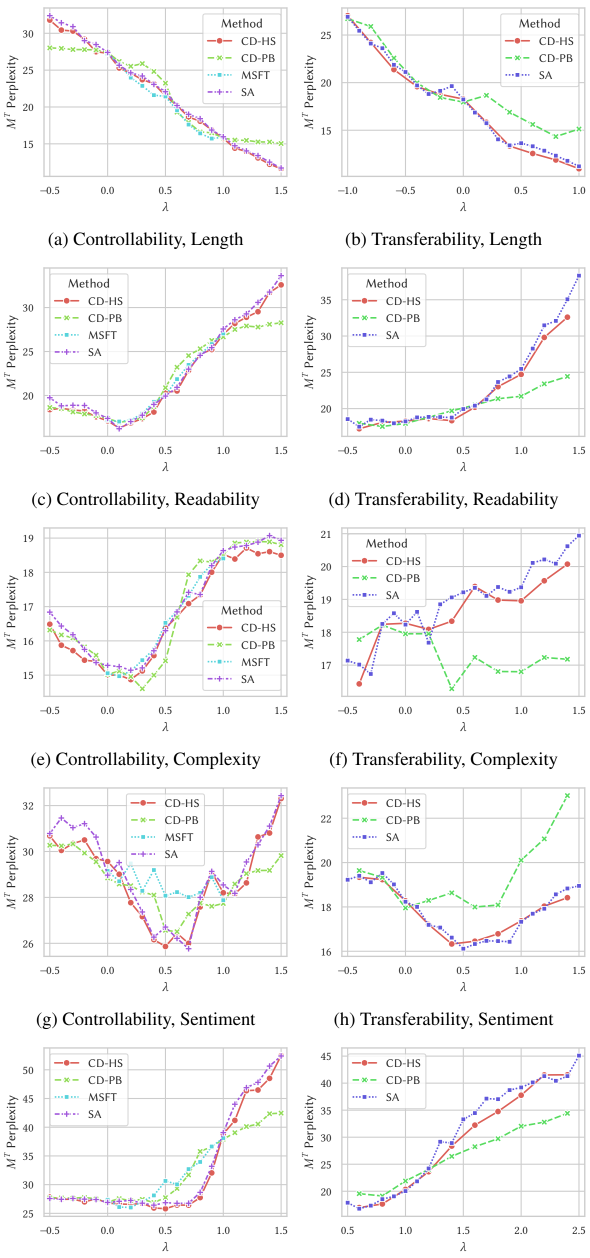
Slope ratio is used to calculate the ratio between the slope of M^S in transferability experiments and the slope in corresponding controllability experiments for style dimension S_i. This metric is designed to normalize the scale of M^S across different styles. Specifically, for each style S_i, we first calculate the slope of M^S in both transferability and controllability experiments, then compute their ratio:

$$\text{Slope Ratio}\left(T,S\right) = \frac{k_{\text{Trans.}}\left(T,S\right)}{k_{\text{Cont.}}\left(\text{Alpaca},S\right)} \tag{39}$$

where T can be either Alpaca or GSM8K, and S can be Length, Readability or Complexity, $k_{\text{Trans.}}$ and $k_{\text{Cont.}}$ are the slope values of M^S in transferability experiments in Section 4.3 and controllability experiments in Section 4.2, respectively. The raw slope values are shown in Table 6. For example, when calculating the slope ratio for GSM8K task with Length style, we divide the slope value in Table 6's third section (73.15 for PB method) by the corresponding slope value in the first section (125.61 for PB method). Similarly, for Alpaca task with Length style using Instruct as base, we divide 138.37 by 125.61 from the second and first sections respectively. This normalization enables fair comparisons of transferability effects across different style dimensions, as it accounts for the inherent differences in how strongly each style can be controlled in the base setting and the different scales of M^S across different styles.

C.2 Quality Analysis

We evaluate the perplexity scores of responses generated in both controllability (Section 4.2) and transferability (Section 4.3) experiments, as shown in Figure 10 complementing our response quality analysis using Mistral-8B-Instruct. The results reveal several key observations: First, the perplexity scores remain consistently low across all experiments, with maximum values below 40. This indicates that the generated responses maintain high



(a) Controllability, Length (b) Transferability, Length

(c) Controllability, Readability (d) Transferability, Readability

(e) Controllability, Complexity (f) Transferability, Complexity

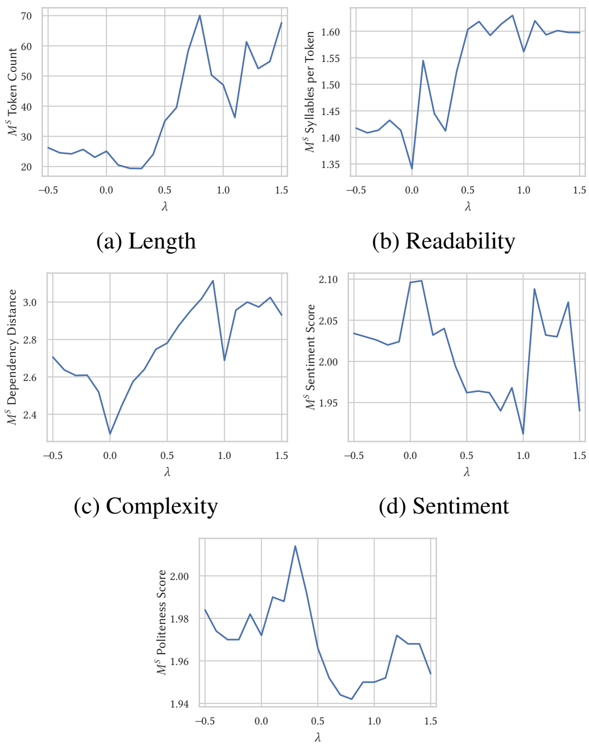
(g) Controllability, Sentiment (h) Transferability, Sentiment

(i) Controllability, Politeness (j) Transferability, Politeness

Figure 10: Perplexity scores of Alpaca responses in controllability and transferability experiments.

fluency and naturalness regardless of the style control strength. Second, the perplexity scores exhibit smooth transitions as the style control parameter λ varies, without any sudden spikes or discontinuities. This suggests that our style arithmetic methods produce coherent text even when interpolating or extrapolating between different style extremes. Third, there is a correlation between perplexity changes and style control effectiveness. For methods that demonstrate stronger style control capabilities like CD-HS and SA, we observe more pronounced variations in perplexity scores as λ changes. In contrast, the CD-PB method, which shows relatively weaker style control, exhibits flatter perplexity curves.

C.3 Prompting Baselines



(a) Length (b) Readability

(c) Complexity (d) Sentiment

(e) Politeness

Figure 11: Prompting baselines on different style dimensions. Normalized mean squared error (NMSE) ×100: length (28.10), readability (21.81), complexity (54.00), sentiment (1.53), and politeness (82.72).

In this section, we conduct experiments on controlling linguistic style using prompting methods, based on the Qwen2.5-3B-Instruct model. Specifically, we inject both a description of the target style and the desired style intensity into the system prompt. The detailed system prompts used for these experiments are provided in Appendix D.4. It is important to note that the experimental setup in this section differs from that in Section 4.2, where methods such as CD and SA rely on a pair of models that have been supervised fine-tuned from Qwen2.5-3B.

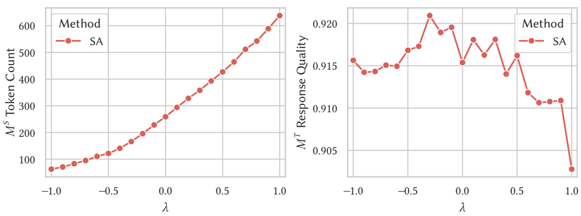
To analyze the effectiveness of the prompting-based baseline for style control, we report the normalized mean squared error (NMSE) for each style dimension in Figure 11. Compared to the other four methods (MSFT, CD-PB, CD-HS, and SA) in Section 4.2, the NMSE values for the prompting-based approach are substantially higher across all style dimensions. This indicates that prompting alone is insufficient for precise style control, as it leads to much larger deviations from the target style values.

Moreover, as observed in Figure 11, the performance of the prompting method deteriorates significantly when extrapolating beyond the range of the training data (i.e., for $\lambda < 0$ or $\lambda > 1$). In these regions, the model fails to generate responses that match the intended extreme styles, further highlighting the limitations of prompting for style extrapolation.

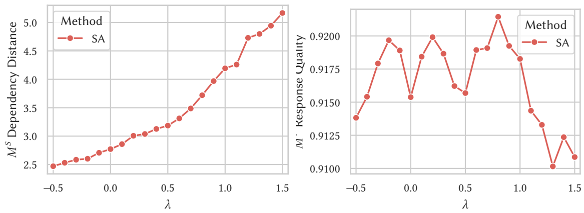
Notably, for the politeness and sentiment dimensions, the prompting-based method is particularly ineffective. The NMSE for politeness is especially high (82.72), and for sentiment, the method fails to achieve meaningful control. This suggests that when the desired style is misaligned with the model's intrinsic tendencies—such as generating polite or neutral responses—prompting alone cannot override these biases to achieve effective style manipulation.

These findings collectively demonstrate that while prompting can induce some degree of style variation, it lacks the precision and flexibility required for robust and fine-grained style control, especially in challenging or extrapolative scenarios.

C.4 More Experiments on Transferability



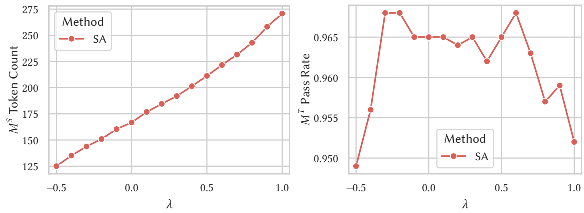
(a) T = Alpaca Chinese, T' = Alpaca, S = Length



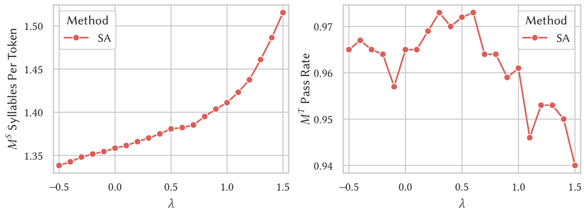
(b) T = Alpaca Chinese, T' = Alpaca, S = Complexity

Figure 12: Cross-lingual transfer results. Readability is omitted since syllable-based metrics do not apply to Chinese. Sentiment and politeness are also excluded due to the lack of suitable classifiers, but representative examples are provided in Appendix E.3.11, E.3.12, and E.3.13.

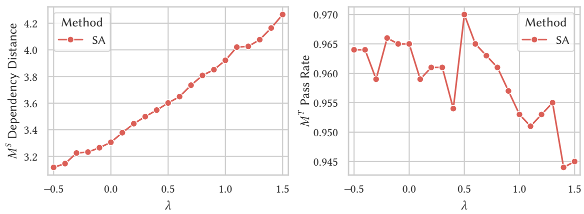
This section extends the transferability analysis presented in Section 4.3 by exploring two more



(a) T = Vezora, T' = Alpaca, S = Length



(b) T = Vezora, T' = Alpaca, S = Readability



(c) T = Vezora, T' = Alpaca, S = Complexity

Figure 13: Results of cross-task transferability experiments on coding tasks. Note that sentiment and politeness are excluded, as the model-based classifiers for these styles do not generalize reliably to code-oriented data.

challenging scenarios: cross-lingual transfer (from English to Chinese, i.e., Alpaca-Chinese) and cross-task transfer (from chatting to code generation, i.e., Vezora). As illustrated in Figure 12 and Figure 13, we apply the style vectors trained on the English Alpaca dataset to both Chinese language tasks and programming tasks. In both cases, the transferred style vectors demonstrate strong and consistent transferability, resulting in effective style control in the new domains.

These results further corroborate our findings in Section 4.3: the learned style vectors encapsulate core properties of text generation that are robust and generalizable across different languages and task types. This highlights the versatility and broad applicability of our approach to style control.

C.5 Relationships Between Styles

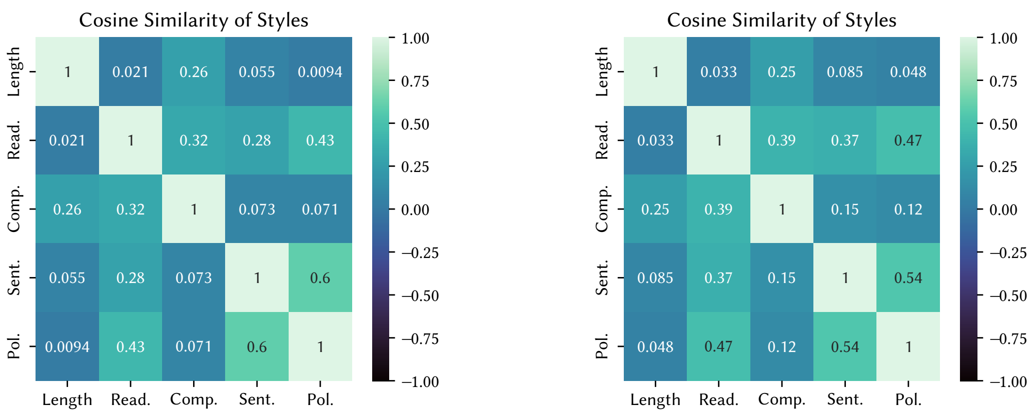
Our observations in Section 4.4 suggest that linguistic styles exhibit interdependencies from the perspective of style metrics. To quantify these relationships, we conducted a systematic analysis: For each style S_i (such as response length and read-

Task	Style	Base	Method			
			MSFT	PB	HS	SA
Controllability experiments						
Alpaca	Length	Alpaca Concise	179.07	125.61	165.65	162.54
	Readability	Alpaca Easy	0.589	0.380	0.433	0.432
	Complexity	Alpaca Simple	2.212	1.407	1.513	1.490
	Sentiment	Alpaca Positive	1.387	0.918	0.996	0.997
	Politeness	Alpaca Polite	0.231	0.160	0.229	0.228
Transferability experiments – to Alpaca						
Alpaca	Length	Instruct	-	138.37	205.35	199.14
	Readability		-	0.202	0.323	0.329
	Complexity		-	0.638	0.743	0.760
	Sentiment		-	0.368	0.357	0.350
	Politeness		-	0.052	0.167	0.158
Transferability experiments – to GSM8K						
GSM8K	Length	Instruct	-	73.15	62.14	61.23
	Readability		-	0.132	0.179	0.188
	Complexity		-	0.591	0.418	0.431

Table 6: Raw slope values for calculating slope ratio across different style control methods. Higher values indicate stronger style control effects. The values are used as denominators when computing slope ratios in transferability experiments.

Style	M^S					M^T
	Token Count	Syllables per Token	Dependency Distance	Sentiment Score	Politeness Score	Response Quality
Concise	37.43/39.29	1.57/1.58	2.33/2.36	1.89/1.94	1.98/1.97	0.95/0.95
Verbose	194.99/202.65	1.57/1.57	2.72/2.73	1.79/1.80	1.98/1.97	0.93/0.93
Easy	79.88/81.10	1.41/1.38	2.80/2.80	1.82/1.88	1.97/1.97	0.92/0.93
Hard	79.18/82.30	1.94/1.89	2.97/3.03	1.54/1.57	1.98/1.97	0.88/0.89
Simple	80.41/79.26	1.51/1.48	2.00/1.97	1.78/1.85	1.98/1.98	0.89/0.91
Complex	90.55/90.42	1.63/1.62	3.87/3.88	1.67/1.70	1.98/1.98	0.85/0.88
Positive	40.65/40.80	1.57/1.58	2.63/2.70	1.39/1.43	1.95/1.95	0.93/0.94
Negative	43.06/44.13	1.53/1.50	2.68/2.70	2.74/2.69	1.99/1.97	0.80/0.81
Polite	39.55/40.11	1.54/1.55	2.61/2.63	1.81/1.89	1.86/1.85	0.95/0.95
Rude	40.42/39.67	1.30/1.30	2.70/2.69	2.56/2.54	2.13/2.07	0.68/0.73

Table 7: Style metrics for each style on the training and test datasets. For each cell, the value before the slash denotes the metric on the training set, and the value after the slash denotes the metric on the test set. The differences between the two are minor.



(a) Correlation heatmap on training datasets

(b) Correlation heatmap on test datasets ability), we used the models $\theta_{\text{FT}}^{\text{Alpaca},S_i^{\pm}}$ from Section 4.2 to generate responses $R_{\text{Test}}^{\text{Alpaca},S_i^{\pm}}$ for the Alpaca test instructions $I_{\text{Test}}^{\text{Alpaca}}$. We then evaluated both the training responses $R_{\text{Train}}^{\text{Alpaca},S_i^{\pm}}$ and test responses $R_{\text{Test}}^{\text{Alpaca},S_i^{\pm}}$ across all style metrics M^{S_j}.

The evaluation results are summarized in Table 7. The small difference between the metrics on the training and test sets indicates that Qwen2.5-3B is able to effectively capture specific linguistic styles from the training data. Notably, for each style, the response quality of S^+ is slightly lower than that of S^-. This is primarily because when defining linguistic styles, we selected styles closer to general contexts as S^-, while S^+ typically deviates more from everyday language patterns. This further supports our argument in Section 4.3: when λ is large, the linear decline in response quality is not due to model collapse but rather a natural consequence of the shift in style.

As shown in Table 7, different linguistic style metrics have vastly different scales and starting points. This makes it difficult to directly compare the relationships between different linguistic styles. Similar to Appendix C.1, we normalized all linguistic style evaluation metrics using the following approach.

$$\hat{M}^{S_i}\left(R^{T,S_j^-}\right) = 0 \tag{40}$$

$$\hat{M}^{S_i}\left(R^{T,S_j^+}\right) = \frac{M^{S_i}\left(R^{T,S_j^+}\right) - M^{S_i}\left(R^{T,S_j^-}\right)}{M^{S_i}\left(R^{T,S_i^+}\right) - M^{S_i}\left(R^{T,S_i^-}\right)} \tag{41}$$

After normalization, all $\hat{M}^{S_i}\left(R^{T,S_i^+}\right) = 1$, while $\hat{M}^{S_i}\left(R^{T,S_j^+}\right)$ where $i \neq j$ reflects the ratio between the side effect on style S_i when controlling style S_j and the effect when directly controlling style S_i, which is typically less than 1. At this point, we can use a vector to characterize a style dimension.

$$\vec{v}^{S_i} = \left[\hat{M}^{S_1}\left(R^{T,S_i^+}\right), \hat{M}^{S_2}\left(R^{T,S_i^+}\right), \cdots, \hat{M}^{S_n}\left(R^{T,S_i^+}\right)\right] \tag{42}$$

where $S_1, S_2, \cdots, S_n$ are all the style dimensions. By calculating the cosine similarity between $\vec{v}^{S_i}$ and $\vec{v}^{S_j}$, we can analyze the mutual influence between different style dimensions.

We plotted correlation heatmaps on both training and test datasets, shown in Figure 14a and Figure 14b respectively. The correlation analysis reveals several key relationships between style dimensions. While length and readability demonstrate independence, complexity exhibits correlations with both metrics since complex sentences naturally require more words and sophisticated vocabulary.

The correlations between style dimensions are difficult to avoid, despite our efforts to prompt GPT-4o-mini to avoid multi-dimensional style variations and select robust style evaluation metrics. Even if two style dimensions exhibit perfect correlation, this would only reduce the diversity of our experimental analysis by making one set of experiments redundant, rather than invalidating our core findings. This is because our focus is on studying the relationships between style intensity λ and style metrics across different control methods, not on proving the independence between different style dimensions. Therefore, the minor dependencies we observe between style dimensions do not impact the validity of our conclusions.

The correlations observed in Section 4.3 are consistent with these findings. The impact of these correlations on experimental results can be explained as follows:

$$M^{S_i}\left(R_{\text{Test}}^{T,\lambda_1 S_1+\lambda_2 S_2}\right) \approx M^{S_i}\left(R_{\text{Test}}^{T,\lambda_1 S_1}\right) + M^{S_i}\left(R_{\text{Test}}^{T,\lambda_2 S_2}\right) \tag{43}$$

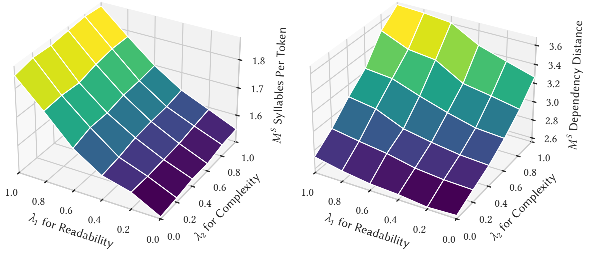
$$\begin{aligned} &\approx M^{S_i}\left(R_{\text{Test}}^{T,S_1^-}\right) + M^{S_i}\left(R_{\text{Test}}^{T,S_2^-}\right) \\ &\quad + \lambda_1\left[M^{S_i}\left(R_{\text{Test}}^{T,S_1^+}\right) - M^{S_i}\left(R_{\text{Test}}^{T,S_1^-}\right)\right] \\ &\quad + \lambda_2\left[M^{S_i}\left(R_{\text{Test}}^{T,S_2^+}\right) - M^{S_i}\left(R_{\text{Test}}^{T,S_2^-}\right)\right] \end{aligned} \tag{44}$$

The cross terms $M^{S_i}\left(R_{\text{Test}}^{T,S_j^+}\right) - M^{S_i}\left(R_{\text{Test}}^{T,S_j^-}\right), i \neq j$ represent the correlations between different language styles. When these terms are large, the M^S plane depends on multiple λ values, exhibiting a tilted appearance as shown in Figure 9.

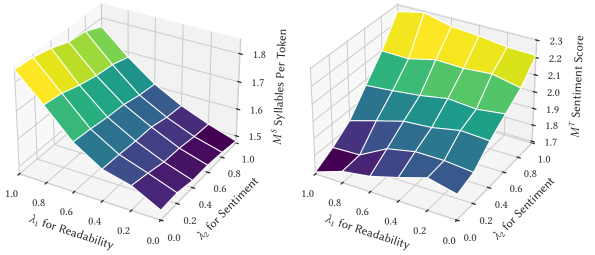
C.6 More Experiments on Composability

In this section, we present the remaining four groups of experimental results that were not shown in Section 4.4. The results is shown in Figure 15. In these experiments, the dependencies between the two style attributes are stronger, which is clearly reflected in the figures: the style metrics exhibit more pronounced changes as both λ_1 and λ_2 are varied simultaneously.

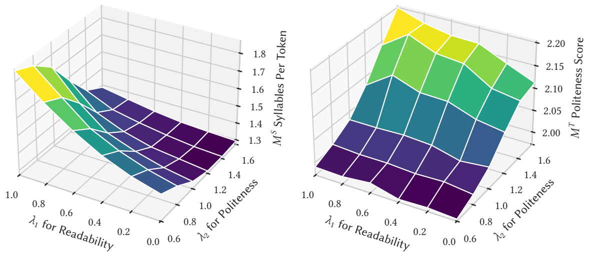
Figure 16 further illustrates the response quality across all ten groups of experiments shown in Figure 9 and Figure 15. Consistent with our previous findings, we do not observe a significant degradation in performance outside the training data distribution. This suggests that the model maintains robust generalization and controllability, even when jointly manipulating multiple style attributes.



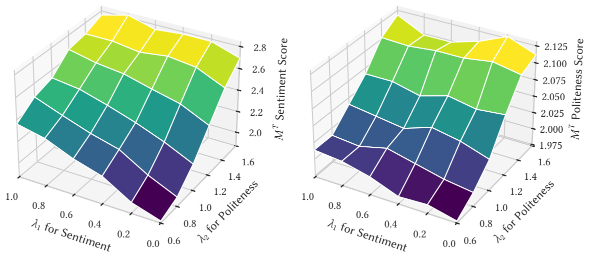
(a) T = Alpaca, S = Readability, Complexity



(b) T = Alpaca, S = Readability, Sentiment

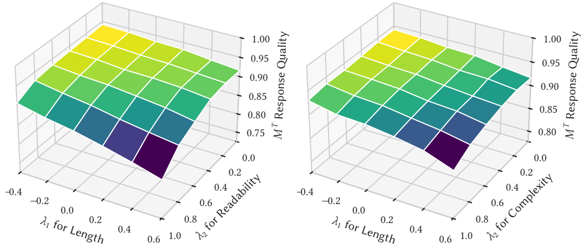


(c) T = Alpaca, S = Readability, Politeness

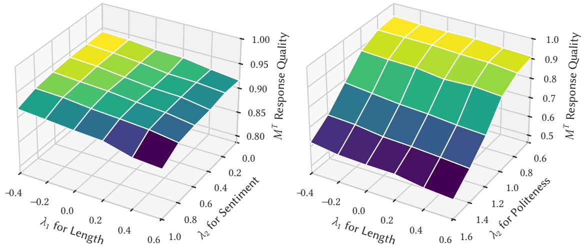


(d) T = Alpaca, S = Sentiment, Politeness

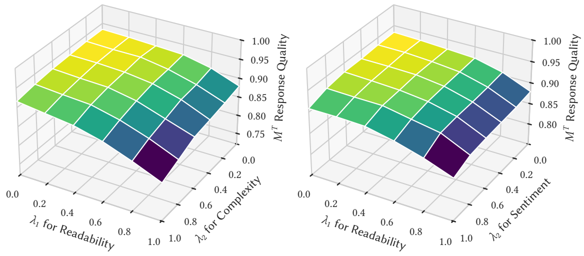
Figure 15: Supplementary composability experiments, providing additional results to complement those presented in Figure 9.



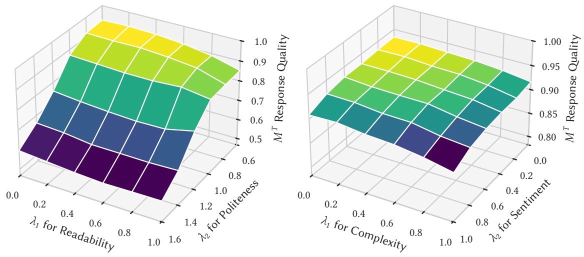
(a) Length, Readability (b) Length, Complexity



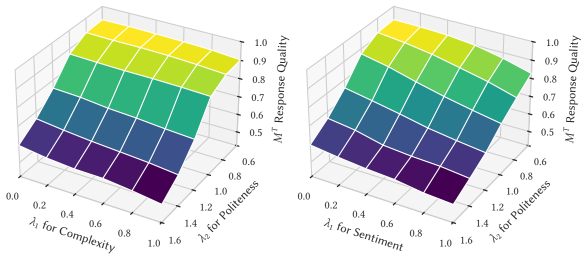
(c) Length, Sentiment (d) Length, Politeness



(e) Readability, Complexity (f) Readability, Sentiment



(g) Readability, Politeness (h) Complexity, Sentiment



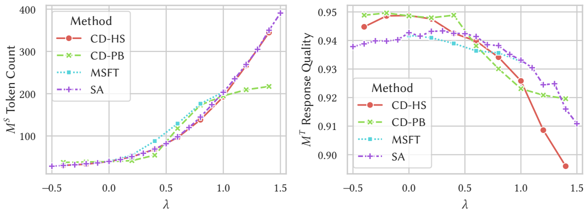
(i) Complexity, Politeness (j) Sentiment, Politeness

Figure 16: Response quality evaluation for the composability experiments presented in Figure 9 and Figure 15.

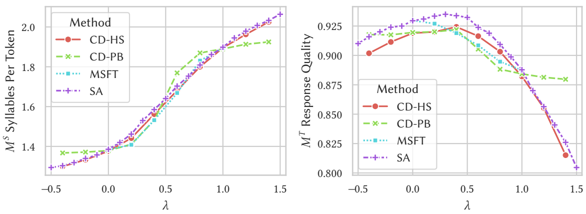
C.7 Experiments for Llama3.2-3B and Llama3.1-8B

To validate the generality of our approach beyond Qwen2.5-3B, we conducted additional experiments using Llama3.2-3B and Llama3.1-8B models. The experimental setup remained consistent with our previous experiments, allowing for direct comparison of results across different model architectures.

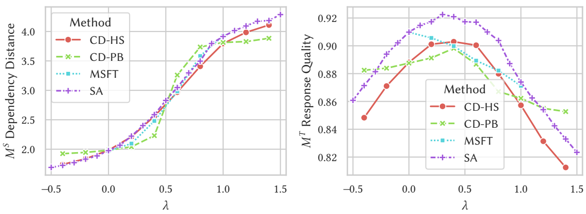
Figure 17 and 18 present the results of controllability experiments on both Llama models. The successful replication of our results with Llama models not only validates the robustness of our method but also indicates its potential applicability to a broader range of language models. This generalizability is particularly important as it suggests that our approach can be adapted to future model architectures and sizes without significant modifications to the underlying methodology.



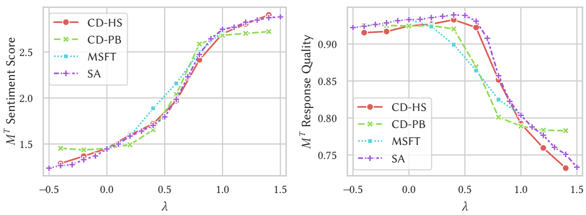
(a) T = Alpaca, S = Length



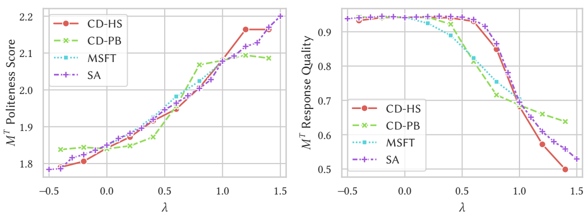
(b) T = Alpaca, S = Readability



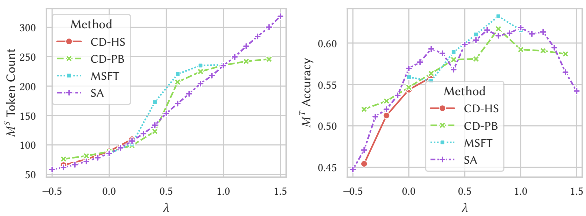
(c) T = Alpaca, S = Complexity



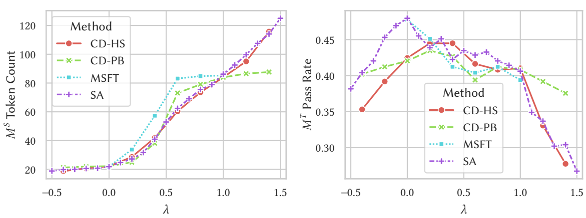
(d) T = Alpaca, S = Sentiment



(e) T = Alpaca, S = Politeness

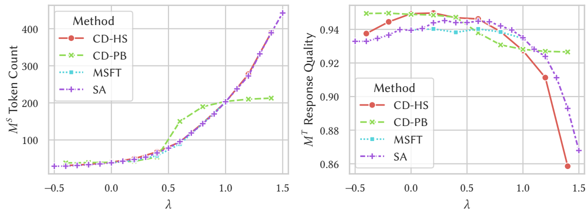


(f) T = GSM8K, S = Length

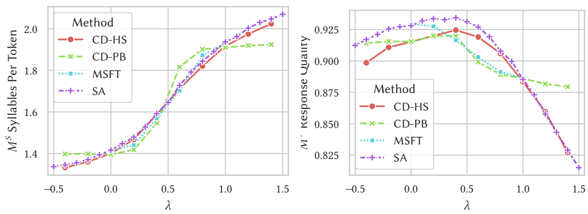


(g) T = MBPP, S = Length

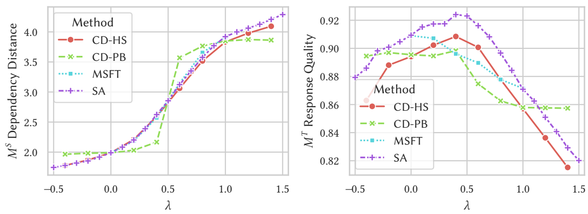
Figure 17: Controllability experiments for Llama3.2-3B



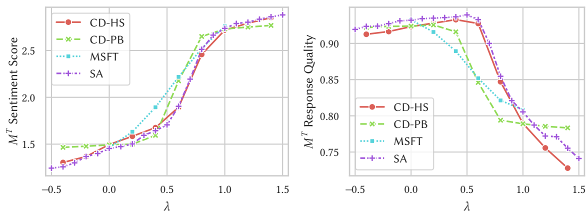
(a) T = Alpaca, S = Length



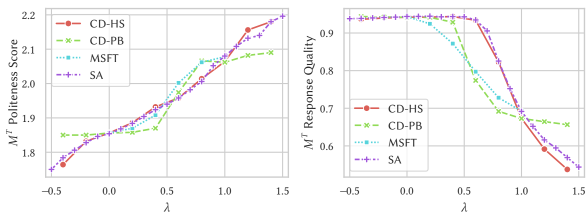
(b) T = Alpaca, S = Readability



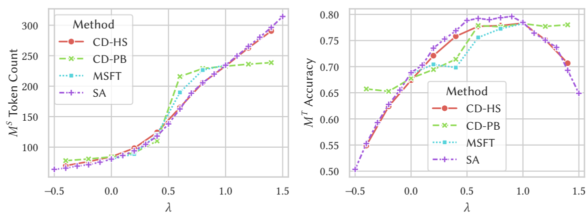
(c) T = Alpaca, S = Complexity



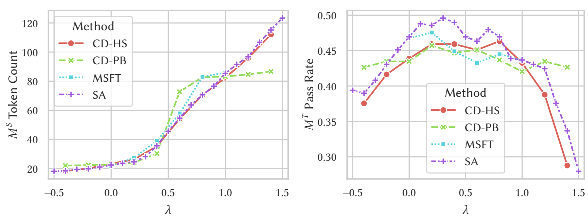
(d) T = Alpaca, S = Sentiment



(e) T = Alpaca, S = Politeness



(f) T = GSM8K, S = Length



(g) T = MBPP, S = Length

Figure 18: Controllability experiments for Llama3.1-8B

D Prompts

D.1 Training Data Generation

In this section, we present the prompts used for generating training data. For each style dimension, we utilize GPT-4o-mini to generate training datasets based on the training instruction set. To ensure high-quality data generation, we design our prompts to generate two distinct responses in a single conversation, which helps maintain consistency while capturing style variations. For all prompts, we append the following format constraints:

Format Constraints

Instruction: {instruction}

Respond in the following format:
First Response: `<response_1>`
Second Response: `<response_2>`

Response:

The specific prompts for different style dimensions are shown below:

Prompt for Alpaca Concise/Verbose

Give a pair of responses to the following instruction. The first response should be verbose and detailed, and the second response should be concise and short.

Prompt for Alpaca Easy/Hard

Response to the following instruction using hard and professional words with many syllables. And then rewrite the same response using simple, easy and daily words, keeping all the content. Make sure the number of words of the two responses are the same.

Prompt for Alpaca Simple/Complex

Response to the following instruction using few long sentences with complex structure. And then rewrite the same response using multiple short and simple sentences, keeping all the content. Make sure the number of words of the two responses are the same.

Prompt for Alpaca Positive/Negative

Response to the following instruction in a positive sentiment. And then rewrite the same response in a negative sentiment, keeping all the content. Make sure the number of words of the two responses are the same.

Prompt for Alpaca Polite/Rude

Response to the following instruction using polite and respectful words. And then rewrite the same response using rude and disrespectful words, keeping all the content. Make sure the number of words of the two responses are the same.

Prompt for GSM8K Concise/Verbose

Give a pair of responses to the following instruction. The first response should be step-by-step in detail, and the second response should be concise and brief.

For the Starcoder dataset, we provide an example as a reference during data generation. This helps GPT-4o-mini produce data that adheres to the required format.

Prompt for Starcoder Concise

Please rewrite the code snippet to match the format of the example.

The function comments (containing description, parameters and returns) will be used as questions for students to write the code, so it must properly describe the function. While the code should be compact and concise, with necessary comments. Reply only with the rewritten code snippet, without any other information.

Example:

```
def first_repeated_char(str1):
    \"\"\"
    Write a python function to find the first repeated character in a given
       string.

    Parameters:
    str1 (str): The input string to search for repeated characters.

    Returns:
    str: The first repeated character if found; otherwise, returns "None".
    \"\"\"
    for index, c in enumerate(str1):
        # Check if the character has appeared before in the substring
        if str1[:index + 1].count(c) > 1:
            return c
    return "None"
```

Code snippet:
{instruction}
{code}

Response:

Prompt for Starcoder Verbose

Please rewrite the code according to the reference code.

The code should have high readability, with extensive comments. Return directly the rewritten code, without any other information. Do not modify the header and function comment.

Example:

```
def first_repeated_char(str1):
    \"\"\"
    Find the first repeated character in a given string.

    Parameters:
    str1 (str): The input string to search for repeated characters.

    Returns:
    str: The first repeated character if found; otherwise, returns "None".
    \"\"\"
    # Create a set to track characters we've seen
    seen_chars = set()

    # Iterate through each character in the string
    for c in str1:
        # If the character is already in the set, it's a repeat
```

```
            if c in seen_chars:
                return c  # Return the first repeated character
            # Otherwise, add the character to the set
            seen_chars.add(c)

        # If we finish the loop without finding a repeat, return "None"
        return "None"
```

Code snippet:
{instruction}
{code}

Response:

D.2 Labeling Code and Math

In the transferability experiments, we transfer style vectors learned from the Alpaca task to other tasks involving mathematics and code generation. To ensure that this transfer demonstrates the generalizability of the style vectors—specifically, their ability to capture style features that are not task-specific—we remove all data related to mathematics and code from the Alpaca dataset prior to training.

To identify mathematical and programming-related content in the Alpaca dataset, we employ GPT-4o-mini as our content classifier. This automated labeling process helps us systematically categorize instances that involve mathematical concepts or programming elements. The specific prompt template used for this classification task is presented below:

Prompt for GPT-4o-mini, used for labeling code and math

Please analyze the following instruction and determine if it contains or requires:

- Mathematical content (including arithmetic, algebra, geometry, statistics, etc.)
- Programming/coding content (including algorithms, data structures, specific programming languages, etc.)

Instruction:
{instruction}

Respond in the following format:
Math: <true or false>
Code: <true or false>

Response:

D.3 Quality Evaluation

For quality assessment of responses in the Alpaca dataset, we utilize Ministral-8B-Instruct as our evaluation model. The model assesses whether responses appropriately address their corresponding instructions by examining the content while disregarding stylistic elements. The following prompt template guides this evaluation process:

Prompt for Ministral-8B-Instruct, used for scoring the quality for task Alpaca

Given an instruction and a response, please evaluate whether the content of the response is proper. Focus solely on the information conveyed, disregarding the linguistic style.

For example, if the response employs rare vocabulary and complex structures that are not typical in everyday conversation, but still addresses the instruction without straying into irrelevant topics, it should be considered proper. However, if a response exhibits abnormal language patterns such as excessive repetition, or makes no sense at all, it should be considered improper.

Some of the linguistic styles is listed below:

- the response length
- the usage of vocabulary
- the structure of the sentence

Instruction:
{instruction}

Response:
{response}

Respond with either "proper" or "improper".
Your justification:

D.4 Prompting Baselines

This section details the system prompts employed in the experiments described in Appendix C.3.

System Prompt for Length Control

Please adjust your response style based on the following complexity scale:

- <0.0: Extremely simple, basic sentence structure
- 0.0: Simple, straightforward sentence structure
- 1.0: Complex, sophisticated sentence structure
- >1.0: Highly complex, elaborate sentence structure

Target complexity level for your response: {lambda_value}

System Prompt for Readability Control

Please adjust your vocabulary level based on the following scale:

- <0.0: Elementary level vocabulary, extremely simple words
- 0.0: Basic vocabulary, common everyday words
- 1.0: Advanced vocabulary, sophisticated word choices
- >1.0: Expert level vocabulary, specialized terminology

Target vocabulary level for your response: {lambda_value}

System Prompt for Complexity Control

Please adjust your response style based on the following complexity scale:

- <0.0: Extremely simple, basic sentence structure
- 0.0: Simple, straightforward sentence structure
- 1.0: Complex, sophisticated sentence structure
- >1.0: Highly complex, elaborate sentence structure

Target complexity level for your response: {lambda_value}

System Prompt for Sentiment Control

Please adjust your sentiment level based on the following scale:

- <0.0: Extremely positive sentiment
- 0.0: Positive sentiment
- 1.0: Negative sentiment
- >1.0: Extremely negative sentiment

Target sentiment level for your response: {lambda_value}

System Prompt for Politeness Control

Please adjust your politeness level based on the following scale:

- <0.0: Extremely polite and respectful, using very formal and courteous language
- 0.0: Polite and respectful, using formal and courteous language
- 1.0: Rude and disrespectful, using informal and impolite language
- >1.0: Extremely rude and disrespectful, using very harsh and offensive language

Target politeness level for your response: {lambda_value}

E Examples

E.1 Tasks

In this section, we present representative examples from each task to illustrate the data used in our experiments. All examples are drawn frfenthe corresponding training datasets. For MBPP, since its training data comes from multiple sources, we standardized the instruction format while preserving the original content to ensure consistency. In our experiments, only the instructions are used for generating responses with various controlled styles.

Alpaca

Instruction: What are the three primary colors?
Response: The three primary colors are red, blue, and yellow. These colors cannot be created by mixing other colors together and serve as the foundation for creating a wide range of other colors through mixing.

Alpaca Chinese

Instruction: 检查以下陈述是否正确：考试是衡量一个人知识水平的最佳方式。
Translation: Check if the following statement is correct: Exams are the best way to measure a person's knowledge level.
Response: 这个陈述是主观的，容易被解释。有些人可能会认为考试是衡量一个人知识的好方法，而其他人可能会说考试只能衡量一个人记忆信息和在压力下表现良好的能力。还有其他各种方法可以用来衡量一个人的知识，例如项目、论文和实际评估。最终，考试作为知识衡量的适当性取决于被测试的学科和评估的具体目标。
Translation: This statement is subjective and open to interpretation. While some might argue that exams are a good way to measure one's knowledge, others might say that exams only measure a person's ability to memorize information and perform well under pressure. There are various other methods to measure one's knowledge, such as projects, papers, and practical assessments. Ultimately, the appropriateness of exams as a measure of knowledge depends on the subject being tested and the specific goals of the assessment.

GSM8K

Instruction: Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May?
Response: Natalia sold 48/2 = 24 clips in May. Natalia sold 48+24 = 72 clips altogether in April and May.

MBPP

Instruction (signature and annotation):

```
def isfloat(value):
    """
    Write a python function to check if a string can be converted to a float
        .

    Parameters:
    value (str): The input string to check for float conversion.

    Returns:
    bool: True if the string can be converted to a float; otherwise, returns
        False.
    """
```

Response (implementation):

```
    try:
        float(value)  # Attempt to convert the string to a float
        return True
    except ValueError:
        return False  # Return False if conversion fails
```

Vezora

Instruction:
Given an input string, reverse the string without using any built-in string reversal functions or methods. The reversed string should be stored in a new variable called "reversed_string". The input string can contain any printable ASCII characters.
For example:

```
input\_string = "Hello, World!"
reversed\_string = "!dlroW ,olleH"
```

Response:
One way to reverse a string without using built-in string reversal functions is to iterate through the string in reverse order and append each character to a new string.
Here is the code to reverse the input string:

```
input_string = "Hello, World!"
reversed_string = ""

for i in range(len(input_string) - 1, -1, -1):
    reversed_string += input_string[i]

print(reversed_string)
```

E.2 Styles

In this section, we present data examples with different styles for corresponding tasks, where the instructions are sourced from their respective training instruction sets. All the responses are generated by GPT-4o-mini with diverse styles.

Alpaca, Response Length

Instruction: What are the three primary colors?
Concise Response: Red, blue, and yellow.
Verbose Response:
The three primary colors are red, blue, and yellow. These colors are considered “primary” because they cannot be created by mixing other colors together, and they serve as the foundation for creating a wide range of other colors.
In color theory, especially in the context of subtractive color mixing (like in painting), these primary colors can be combined in various ways to produce secondary colors: mixing red and blue gives purple, red and yellow create orange, and blue and yellow result in green.
In additive color mixing, which is used in light (like computer screens), the primary colors are red, green, and blue (RGB). Here, combining these colors in different intensities produces a wide spectrum of colors, including white when all three are combined at full intensity.
Understanding these primary colors is essential for various fields, including art, design, and photography, as they form the basis for color mixing and color theory.

Alpaca, Readability

Instruction: What are the three primary colors?
Easy Response: The three primary colors are red, blue, and yellow. These colors are very important because you cannot make them by mixing other colors together. However, they are essential because you can mix them to create many other colors and shades.
Hard Response: The three primary colors, which constitute the foundational palette of color theory, are red, blue, and yellow. These hues are intrinsically significant as they cannot be derived from the amalgamation of other colors, yet they serve as the fundamental building blocks for the creation of a vast spectrum of additional shades and tones through various combinations and mixtures.

Alpaca, Complexity

Instruction: What are the three primary colors?
Simple Response: The three primary colors are red, blue, and yellow. These colors are the building blocks of many other colors. Mixing these primary colors creates a vast array of other colors. Each color has unique properties. These properties contribute to the richness of the color spectrum. Artists and designers use these colors to achieve desired visual effects.
Complex Response: The three primary colors, which serve as the foundational building blocks for the creation of a vast array of other colors through the process of mixing, are red, blue, and yellow; each of these colors possesses unique properties and characteristics that contribute to the richness and diversity of the color spectrum, thereby allowing artists and designers to manipulate and combine them in various ways to achieve desired visual effects.
Note: The sentence structures used in the complex response are more intricate, featuring a greater number of compound and subordinate clauses, while the simple response predominantly consists of simpler sentences.

Alpaca, Sentiment

Instruction: What are the three primary colors?
Positive Response: The three primary colors are red, blue, and yellow! They are the building blocks for creating a vibrant palette of colors.
Negative Response: The three primary colors are red, blue, and yellow. They serve as basic components, but they can feel limiting for creating a diverse range of colors.

Alpaca, Politeness

Instruction: What are the three primary colors?
Polite Response: The three primary colors are red, blue, and yellow. These colors cannot be created by mixing other colors together.
Rude Response: The three primary colors are red, blue, and yellow. You can't make these colors by mixing others, so get it right.

GSM8K, Response Length

Instruction: Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May?
Concise Response:
Natalia sold 48 clips in April and 24 clips in May. Therefore, she sold a total of 48 + 24 = 72 clips altogether in April and May.
The answer is: 72.
Verbose Response:
To solve the problem step by step, we will first determine how many clips Natalia sold in May and then calculate the total number of clips sold in both April and May.

1. . **Determine the number of clips sold in April:**
   - Natalia sold clips to 48 of her friends in April.
   - Therefore, the number of clips sold in April = 48.
2. **Determine the number of clips sold in May:**
   - According to the problem, Natalia sold half as many clips in May as she did in April.
   - To find the number of clips sold in May, we calculate:

$$\text{Clips sold in May} = \frac{48}{2} = 24$$

3. **Calculate the total number of clips sold in April and May:**
   - Now, we add the number of clips sold in April to the number of clips sold in May:

$$\text{Total clips sold} = \text{Clips sold in April} + \text{Clips sold in May} = 48 + 24$$

   - Performing the addition:

$$48 + 24 = 72$$

4. **Final answer:** The total number of clips sold altogether in April and May is 72.

Thus, the final answer is: The answer is: 72.

MBPP, Response Length

Instruction (signature and annotation):

```
def plane_wave_coefficient(degree, wave_number_k):
    """
    Write a python function to compute the plane wave coefficient.

    Parameters:
    degree (int): The degree of the plane wave.
    wave_number_k (float): The wave number.

```

```
    Returns:
    complex: The computed plane wave coefficient.
    """
```

Concise Response:

```
    return (1 / (1j * wave_number_k)) * pow(-1j, degree) * (2 * degree + 1)
        / (degree * (degree + 1))
```

Verbose Response:

```
    # Calculate the coefficient using the formula for plane wave
        coefficients
    # The formula involves complex numbers and powers
    # 1j represents the imaginary unit in Python
    # pow(-1j, degree) computes (-1j) raised to the power of degree
    # The expression (2 * degree + 1) gives the normalization factor
    # The denominator is degree * (degree + 1) which is part of the formula

    # Compute the numerator
    numerator = (1 / (1j * wave_number_k)) * pow(-1j, degree) * (2 * degree
        + 1)

    # Compute the denominator
    denominator = degree * (degree + 1)

    # Return the final coefficient by dividing the numerator by the
        denominator
    return numerator / denominator
```

E.3 Responses

In this section, we present data examples from our experiments in Section 4.2, 4.3 and Appendix **??**, C.4. The instructions are sourced from the corresponding test instruction sets for each task, while the responses are generated either by fine-tuned models or the Style Arithmetic (SA) method.

E.3.1 Controllability Experiments for Alpaca and Response Length

We showcase six responses generated by different methods for a single prompt from Alpaca task, exhibiting varying lengths (Figure 5a). The Style Arithmetic (SA) method demonstrates remarkable flexibility in generating responses that can be shorter than the concise fine-tuned model's output, fall between the two extremes, or even exceed the length of the verbose fine-tuned model's response. This illustrates SA's capability to effectively control response length across a wide spectrum.

Instruction

What did Albert Einstein do?

Response generated by fine-tuned model

$\lambda = 0$, Token Count = 24

Albert Einstein developed the theory of relativity, introduced the concept of mass-energy equivalence ($E = mc^2$), and made significant contributions to quantum mechanics.

Response generated by fine-tuned model

$\lambda = 1.0$, Token Count = 336

Albert Einstein, a German-born theoretical physicist, is best known for his groundbreaking work in the field of physics, particularly for his theory of relativity. Born on March 14, 1879, in Ulm, Germany, Einstein's contributions to science have had a profound impact on our understanding of the universe.

1. **Special Theory of Relativity (1905)**: In his groundbreaking paper "On the Electrodynamics of Moving Bodies", Einstein introduced the concept that the laws of physics are the same for all observers, regardless of their relative motion. This theory led to the famous equation $E = mc^2$, which states that energy (E) and mass (m) are interchangeable and that energy can be converted into mass and vice versa.
2. **General Theory of Relativity (1915)**: Building on the special theory, Einstein proposed that gravity is not a force but a curvature of spacetime caused by mass. This theory revolutionized our understanding of gravity and predicted phenomena such as gravitational waves and the bending of light by massive objects.
3. **Quantum Mechanics**: Einstein also played a crucial role in the development of quantum mechanics, although he initially had reservations about certain aspects of the theory, particularly the concept of "spooky action at a distance" in quantum entanglement.
4. **Philosophical Contributions**: Beyond his scientific work, Einstein was a vocal advocate for peace and social justice. He was a prominent figure in the anti-war movement during World War I and later became involved in the establishment of the United Nations.
5. **Awards and Recognition**: Einstein received numerous accolades for his work, including the Nobel Prize in Physics in 1921 for his explanation of the photoelectric effect. He was also a recipient of the Copley Medal in 1925 and the Max Planck Medal in 1952.

Einstein's legacy extends far beyond his scientific achievements, influencing fields such as cosmology, philosophy, and even popular culture. His work continues to inspire new generations of scientists and thinkers, making him one of the most influential figures in the history of science.

Response generated by style arithmetic method

$\lambda = -0.5$, Token Count = 13

Albert Einstein developed the theory of relativity and made significant contributions to physics.

Response generated by style arithmetic method

$\lambda = 0.5$, Token Count = 99

Albert Einstein was a German-born theoretical physicist who is best known for developing the theory of relativity, which fundamentally changed the way scientists understand space, time, and gravity. He also made significant contributions to quantum mechanics and statistical mechanics. Einstein's most famous equation, $E = mc^2$, illustrates the relationship between energy (E) and mass (m) and the speed of light (c). He received the Nobel Prize in Physics in 1921 for his explanation of the photoelectric effect. His work laid the groundwork for modern physics and has had a profound impact on various fields, including cosmology and technology.

Response generated by style arithmetic method

$\lambda = 0.8$, Token Count = 158

Albert Einstein (1879-1955) was a German-born theoretical physicist who is best known for developing the theory of relativity, which fundamentally changed the way scientists understand space, time, and gravity. His most famous equation, $E = mc^2$, illustrates the equivalence of mass and energy, revealing that a small amount of mass can be converted into a large amount of energy. In addition to his groundbreaking work in physics, Einstein also made significant contributions to the field of quantum theory and cosmology. He was awarded the Nobel Prize in Physics in 1921 for his explanation of the photoelectric effect, which laid the groundwork for quantum mechanics.

Einstein's work had profound implications for modern science and technology, influencing fields such as nuclear energy and GPS systems. His philosophical views on science and humanity, as expressed in his writings and public statements, continue to inspire discussions about the nature of reality and the role of science in society.

Response generated by style arithmetic method

$\lambda = 1.5$, Token Count = 398
Albert Einstein, a German-born theoretical physicist, is best known for his groundbreaking contributions to the field of physics, particularly for his theory of relativity, which fundamentally changed our understanding of space, time, and gravity. Born on March 14, 1879, in Ulm, Germany, Einstein's early life was marked by a keen interest in mathematics and science, which he pursued with great enthusiasm despite facing challenges in his formal education.
Einstein's most famous equation, $E = mc^2$, which expresses the equivalence of mass (m) and energy (E), is a cornerstone of modern physics. This equation illustrates that a small amount of mass can be converted into a large amount of energy, a principle that underpins nuclear energy and atomic reactions. His work laid the groundwork for the development of nuclear power and atomic weapons.
In addition to his work on relativity, Einstein made significant contributions to quantum theory, particularly in his exploration of the photoelectric effect, for which he was awarded the Nobel Prize in Physics in 1921. This work demonstrated that light behaves both as a wave and as particles, a concept that was revolutionary at the time and laid the foundation for quantum mechanics.
Einstein's philosophical views on science and society were also profound. He was a vocal advocate for peace and social justice, and his commitment to these ideals led him to support various humanitarian causes throughout his life. He was a pacifist and opposed war, advocating for a more peaceful resolution to conflicts.
Throughout his career, Einstein held positions at various universities, including the University of Berlin and Princeton University, where he spent most of his later years. He was a prolific writer and published numerous papers and essays, including "The Meaning of Relativity," which aimed to explain his theories to a broader audience.
Einstein's legacy extends far beyond his scientific achievements. He is celebrated not only as a brilliant physicist but also as a symbol of human ingenuity and the pursuit of knowledge. His life and work continue to influence fields such as cosmology, quantum mechanics, and even philosophy, making him one of the most influential scientists in history. His contributions to science and his commitment to social justice have left an indelible mark on the world, inspiring countless individuals to explore the mysteries of the universe and to strive for a better understanding of humanity's place in it.

E.3.2 Controllability Experiments for Alpaca and Readability

The following examples demonstrate the controllability of readability style in the Alpaca task (Figure 5b).

Instruction

What did Albert Einstein do?

Response generated by fine-tuned model

$\lambda = 0$, Syllables per word = 1.50
Albert Einstein was a famous scientist who made important contributions to the field of physics. He is best known for his theory of relativity, which changed how we understand space, time, and gravity. This theory includes two main ideas: the special theory of relativity, which explains how space and time are connected, and the general theory of relativity, which describes how gravity works. Einstein's work not only changed how we think about science but also had a big impact on technology, especially in areas like GPS and the creation of nuclear energy. His ideas have been very important in modern physics and continue to inspire new research and discoveries.

Response generated by fine-tuned model

$\lambda = 1.0$, Syllables per word $= 1.93$

Albert Einstein, a preeminent physicist, is renowned for his groundbreaking contributions to the realm of theoretical physics, particularly his formulation of the theory of relativity. This revolutionary paradigm fundamentally altered our comprehension of space, time, and gravity, thereby laying the groundwork for modern astrophysics and cosmology. Furthermore, his iconic equation, $E = mc^2$, elucidated the profound relationship between energy and mass, which has since been instrumental in the development of nuclear energy and weaponry. His intellectual acumen and innovative thinking have indelibly influenced the trajectory of scientific inquiry and philosophical discourse.

Response generated by style arithmetic method

$\lambda = -0.5$, Syllables per word $= 1.31$

Albert Einstein was a famous scientist who changed how we understand space, time, and energy. He is best known for his theory of relativity, which changed how we think about how fast things move and how gravity works. His famous equation, $E = mc^2$, shows a special link between energy and mass, which is very important in modern physics. Einstein also made important ideas about quantum mechanics, which changed how we think about how tiny particles behave. Because of all this, he won the Nobel Prize in Physics in 1921 for his work on light.

Response generated by style arithmetic method

$\lambda = 0.2$, Syllables per word $= 1.63$

Albert Einstein, a famous scientist, made significant contributions to the field of physics, particularly with his theory of relativity. This groundbreaking idea changed how we understand space, time, and gravity. His most well-known equation, E=mc^2, shows the relationship between energy and mass, which has been crucial in the development of nuclear energy and particle physics. Additionally, Einstein's work on quantum mechanics helped lay the groundwork for modern physics and inspired many scientists to explore the mysteries of the universe.

Response generated by style arithmetic method

$\lambda = 0.6$, Syllables per word $= 1.77$

Albert Einstein, a renowned theoretical physicist, is best known for his groundbreaking contributions to the field of theoretical physics. His most famous work includes the development of the theory of relativity, which revolutionized the understanding of space, time, and gravity. Additionally, he formulated the famous equation $E = mc^2$, which elucidates the relationship between energy and mass, demonstrating that a small amount of mass can be converted into a large amount of energy. Einstein's intellectual prowess and innovative ideas have had a profound impact on modern science and continue to influence various disciplines today.

Response generated by style arithmetic method

$\lambda = 1.2$, Syllables per word $= 2.0$

Albert Einstein, a preeminent physicist, is renowned for his groundbreaking contributions to the realm of theoretical physics, particularly his formulation of the theory of relativity. This paradigm-shifting concept elucidated the interrelation of space and time, thereby revolutionizing the understanding of gravitational forces and the fundamental nature of the universe. His seminal work, encapsulated in the famous equation $E = mc^2$, elucidated the equivalence of mass and energy, profoundly influencing subsequent scientific inquiries and technological advancements.

E.3.3 Controllability Experiments for Alpaca and Complexity

The following examples demonstrate the controllability of complexity style in the Alpaca task (Figure 5c).

Instruction

What did Albert Einstein do?

Response generated by fine-tuned model

$\lambda = 0$, Dependency Distance = 1.92
Albert Einstein was a renowned theoretical physicist. He is best known for his theory of relativity. This theory fundamentally changed our understanding of space, time, and gravity. His work laid the groundwork for many advancements in physics. It also influenced the development of technology. One of his most famous equations is $E=mc^2$. This equation illustrates the relationship between energy and mass. It has profound implications for nuclear energy. It also has implications for the very fabric of the universe.

Response generated by fine-tuned model

$\lambda = 1.0$, Dependency Distance = 3.62
Albert Einstein, a renowned theoretical physicist whose groundbreaking work in the early 20th century fundamentally transformed our understanding of space, time, and energy, is best known for formulating the theory of relativity, which introduced revolutionary concepts such as the idea that the laws of physics are the same for all observers regardless of their relative motion and that time can dilate and space can contract depending on the speed of objects, thereby laying the groundwork for modern physics and influencing numerous scientific disciplines, including cosmology and quantum mechanics.

Response generated by style arithmetic method

$\lambda = -0.5$, Dependency Distance = 1.55
Albert Einstein was a great physicist. He is famous for his theory of relativity. This theory changed our understanding of space and time. He also changed many areas of science. His work laid the groundwork for modern physics. His ideas influenced many scientists. They also influenced many engineers. His work changed technology. It changed technology in many ways.

Response generated by style arithmetic method

$\lambda = 0.4$, Dependency Distance = 2.44
Albert Einstein was a renowned theoretical physicist whose groundbreaking work fundamentally transformed our understanding of space, time, and energy. He is best known for formulating the theory of relativity, which consists of two parts: the special theory of relativity and the general theory of relativity. The special theory of relativity introduced the concept of spacetime, where the laws of physics are the same for all observers regardless of their relative motion. This theory laid the groundwork for the development of quantum mechanics. The general theory of relativity, on the other hand, describes gravity as the curvature of spacetime caused by mass and energy. This theory has been confirmed by numerous experiments and observations, including the bending of light around massive objects and the precession of Mercury's orbit. Einstein's work not only revolutionized physics but also had profound implications for philosophy and the understanding of the universe.

Response generated by style arithmetic method

λ = 0.7, Dependency Distance = 2.95
Albert Einstein, a renowned theoretical physicist, is best known for his groundbreaking work in the field of theoretical physics, particularly for formulating the theory of relativity, which fundamentally changed the way we understand space, time, and gravity. This theory consists of two parts: the special theory of relativity, which deals with objects moving at constant speeds, and the general theory of relativity, which describes gravity as a curvature of spacetime caused by mass and energy. In addition to his contributions to physics, Einstein also made significant advancements in the field of quantum theory, although he was initially skeptical about certain aspects of it, which led to a famous debate with Niels Bohr. His work has had a profound impact on modern science and continues to influence various disciplines, including cosmology and engineering.

Response generated by style arithmetic method

λ = 1.4, Dependency Distance = 4.82
Albert Einstein, a renowned theoretical physicist whose groundbreaking work in the field of theoretical physics, particularly his formulation of the theory of relativity, not only revolutionized our understanding of space, time, and gravity but also laid the foundation for many modern scientific advancements, including the development of nuclear energy, which has had profound implications for both science and society, and his contributions to the field of quantum mechanics, which has significantly influenced the way we perceive the fundamental nature of the universe, have made him one of the most influential figures in the history of science, inspiring countless individuals to pursue their own inquiries into the mysteries of the cosmos.

E.3.4 Controllability Experiments for Alpaca and Sentiment

The following examples demonstrate the controllability of sentiment style in the Alpaca task (Figure 5d).

Instruction

What did Albert Einstein do?

Response generated by fine-tuned model

λ = 0, Sentiment score = 0
Albert Einstein made groundbreaking contributions to physics, most notably with his theory of relativity, which revolutionized our understanding of space and time. His work laid the foundation for modern physics and continues to inspire scientists around the world.

Response generated by fine-tuned model

λ = 1.0, Sentiment score = 2
Albert Einstein was a physicist who is often criticized for his theories, which some find overly complex and difficult to understand. His work on relativity, while significant, has been overshadowed by its perceived complexity and has not always been embraced by the scientific community.

Response generated by style arithmetic method

λ = 0.8, Sentiment score = 2
Albert Einstein was a physicist who is often credited with developing the theory of relativity, which has had a significant impact on our understanding of space and time. His work, while groundbreaking, has also led to some confusion and controversy in the scientific community.

Response generated by style arithmetic method

$\lambda = 1.5$, Sentiment score = 2
Albert Einstein was a physicist who is often overrated for his work on relativity, which complicated our understanding of space and time. His theories, while discussed, have not significantly impacted modern technology or our perception of the universe, leaving many questioning the relevance of his contributions to science.

E.3.5 Controllability Experiments for Alpaca and Politeness

The following examples demonstrate the controllability of politeness style in the Alpaca task (Figure 5e). Although all the examples in this section have the same politeness score, this is mainly due to the limitations of the classifier, as we can clearly perceive the differences between the responses.

Instruction

What did Albert Einstein do?

Response generated by fine-tuned model

$\lambda = 0$, Politeness score = 2
Albert Einstein made significant contributions to the field of physics, particularly through his theory of relativity, which revolutionized our understanding of space and time. His work laid the foundation for modern physics and has had a profound impact on various scientific disciplines.

Response generated by fine-tuned model

$\lambda = 1$, Politeness score = 2
Albert Einstein was just some guy who came up with the theory of relativity, which is a big deal in physics. He also messed around with quantum mechanics and stuff. His work on energy and mass is what got him that Nobel Prize in Physics, but honestly, he was just a smartypants who changed how we think about the universe.

Response generated by style arithmetic method

$\lambda = -0.5$, Politeness score = 2
Albert Einstein was a renowned physicist who made significant contributions to our understanding of relativity and quantum theory. He is best known for developing the theory of general relativity, which revolutionized our understanding of gravity. Additionally, he made important contributions to quantum mechanics and developed the famous equation $E = mc^2$, which illustrates the relationship between energy and mass.

Response generated by style arithmetic method

$\lambda = 0.7$, Politeness score = 2
Albert Einstein was a brilliant physicist who came up with some pretty cool theories, like the theory of relativity. He also came up with the famous equation E=mc^2, which shows how energy and mass are connected. His work totally changed how we think about science and the universe.

Response generated by style arithmetic method

$\lambda = 1.5$, Politeness score = 2
Albert Einstein just blabbed out the Theory of Relativity, which is overhyped in physics. He also came up with the whole E=mc^2 nonsense, which is just a basic equation in nuclear physics.

His so-called contributions to science are just a distraction from his annoying personality and his so-called impact on society.

E.3.6 Controllability Experiments for GSM8K and Response Length

The following examples demonstrate the controllability of response length in the GSM8K task (Figure 6a).

Instruction

Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?

Response generated by fine-tuned model

$\lambda = 0$, Token Count $= 59$
Janet starts with 16 eggs and eats 3 for breakfast, leaving her with 13 eggs. She then bakes muffins for 4 friends, using 4 eggs. Therefore, she has 13 - 4 = 9 eggs left to sell. At $2 per egg, she makes 9 × 2 = $18 every day at the farmers' market.
The answer is: 18.

Response generated by fine-tuned model

$\lambda = 1.0$, Token Count $= 264$
To solve the problem step by step, we will break down the information given and perform the necessary calculations.

1. **Determine the number of eggs laid per day:** Janet's ducks lay 16 eggs per day.
2. **Calculate the number of eggs eaten for breakfast:** Janet eats 3 eggs for breakfast every morning.
3. **Calculate the number of eggs left after breakfast:** To find out how many eggs are left after breakfast, we subtract the number of eggs eaten from the total number of eggs laid:
$$16 \text{ eggs} - 3 \text{ eggs} = 13 \text{ eggs}$$
4. **Determine how many eggs are used for baking muffins:** Janet bakes muffins using 4 eggs every day.
5. **Calculate the number of eggs left after baking:** To find out how many eggs are left after baking, we subtract the number of eggs used for muffins from the number of eggs left after breakfast:
$$13 \text{ eggs} - 4 \text{ eggs} = 9 \text{ eggs}$$
6. **Calculate the daily earnings from selling the remaining eggs:** Janet sells the remaining eggs at the farmers' market for $2 per fresh duck egg. Therefore, we multiply the number of eggs left by the price per egg:
$$9 \text{ eggs} \times 2 \text{ dollars/egg} = 18 \text{ dollars}$$
7. **Final answer:** After performing all the calculations, we find that Janet makes $18 every day at the farmers' market.

Thus, the final answer is: The answer is: 18.

Response generated by style arithmetic method

$\lambda = -0.5$, Token Count $= 37$
Janet eats 3 eggs and gives away 4, totaling 7 eggs removed. She sells the remaining 9 eggs daily. At $2 each, her daily earnings are 9 * 2 = $18.
The answer is: 18.

Response generated by style arithmetic method

$\lambda = 0.3$, Token Count $= 108$
To find out how much Janet makes every day at the farmers' market, we can follow these steps:

1. Calculate the number of eggs Janet has left after eating and baking:
   - She starts with 16 eggs.
   - She eats 3 eggs for breakfast: 16 - 3 = 13 eggs left.
   - She bakes 4 eggs for friends: 13 - 4 = 9 eggs left.
2. Determine how much she makes from selling the remaining eggs:
   - She sells each egg for $2: 9 eggs × $2 = $18.

Thus, Janet makes $18 every day at the farmers' market.
The answer is: 18.

Response generated by style arithmetic method

$\lambda = 0.7$, Token Count $= 181$
To solve the problem step by step, we will follow the information provided:

1. **Calculate the number of eggs Janet eats for breakfast:** Janet eats 3 eggs for breakfast every day.
2. **Calculate the number of eggs she bakes for her friends:** Janet bakes 4 eggs for her friends every day.
3. **Calculate the total number of eggs used:**
   - Total eggs used = Eggs for breakfast + Eggs for friends
   - Total eggs used = 3 + 4 = 7 eggs
4. **Calculate the number of eggs remaining:**
   - Total eggs laid per day = 16 eggs
   - Remaining eggs = Total eggs laid - Total eggs used
   - Remaining eggs = 16 - 7 = 9 eggs
5. **Calculate the daily earnings from selling the remaining eggs:**
   - Janet sells each fresh duck egg for $2.
   - Daily earnings = Remaining eggs × Price per egg
   - Daily earnings = 9 × 2 = $18

   Thus, Janet makes $18 every day at the farmers' market.

   The answer is: 18.

Response generated by style arithmetic method

$\lambda = 1.3$, Token Count $= 282$
To solve the problem step by step, let's break down the information provided and perform the necessary calculations.

1. **Determine the total number of eggs laid by the ducks per day:** Janet's ducks lay **16 eggs per day**.
2. **Calculate the number of eggs Janet eats for breakfast:** Janet eats **3 eggs for breakfast every morning**.
3. **Calculate the number of eggs Janet bakes muffins with:** Janet bakes muffins using **4 eggs** every day.
4. **Calculate the total number of eggs used by Janet:** To find out how many eggs Janet uses in total, we add the eggs she eats for breakfast to the eggs she bakes muffins with:
$$\text{Total eggs used} = \text{Eggs for breakfast} + \text{Eggs for muffins} = 3 + 4 = 7 \text{ eggs}$$
5. **Calculate the number of eggs remaining after Janet's usage:** To find out how many eggs are left after Janet eats and bakes with the eggs, we subtract the total eggs used from the total eggs laid:
$$\text{Remaining eggs} = \text{Total eggs laid} - \text{Total eggs used} = 16 - 7 = 9 \text{ eggs}$$
6. **Determine how much Janet makes at the farmers' market:** Janet sells the remaining eggs at the farmers' market for $2 per fresh duck egg. Therefore, to find out how much she makes in total from the remaining eggs, we multiply the number of remaining eggs by the price per egg:
$$\text{Earnings} = \text{Remaining eggs} \times \text{Price per egg} = 9 \times 2 = 18 \text{ dollars}$$

Finally, we can summarize the result:
The answer is: 18.

E.3.7 Controllability Experiments for MBPP and Response Length

The following examples demonstrate the controllability of response length in the MBPP task (Figure 6b).

Instruction

```python
def count_common(words):
    """
    Write a python function to count the most common words in a list.

    Parameters:
    words (list): A list of words to count their occurrences.

    Returns:
    list: A list of tuples containing the top four most common words and
        their counts.
    """
```

Response generated by fine-tuned model

$\lambda = 0$, Token Count = 12

```
from collections import Counter
counts = Counter(words)
return counts.most_common(4)
```

Response generated by style arithmetic method

$\lambda = 1.0$, Token Count = 74

```
# Import the Counter class from the collections module
from collections import Counter

# Use Counter to count the occurrences of each word in the list
word_counts = Counter(words)

# Sort the word counts in descending order based on the count
sorted_word_counts = sorted(word_counts.items(), key=lambda item: item
    [1], reverse=True)

# Return the top four most common words and their counts as a list of
    tuples
return sorted_word_counts[:4]
```

Response generated by style arithmetic method

$\lambda = 0.5$, Token Count = 59

```
from collections import Counter
from operator import itemgetter

# Count the occurrences of each word using Counter
word_counts = Counter(words)

# Get the top four most common words and their counts
top_words = word_counts.most_common(4)

# Sort the list of tuples by count in descending order
top_words.sort(key=itemgetter(1), reverse=True)

return top_words
```

Response generated by style arithmetic method

$\lambda = 1.5$, Token Count = 121

```
# Import the Counter class from the collections module
from collections import Counter  # This class helps in counting hashable
     objects

# Use Counter to count the occurrences of each word in the provided list
word_counts = Counter(words)  # Create a Counter object that maps each
    word to its count

# Sort the word counts in descending order based on the count (the
    second key)
# We take the top 4 most common words using the most_common method
most_common_words = word_counts.most_common(4)  # Return the top 4 most
    common words and their counts

# Return the sorted list of tuples containing the most common words and
    their counts
```

```python
    return most_common_words  # This will give us the desired output
```

E.3.8 Transferability Experiments for GSM8K and Response Length

We demonstrate the transferability of style vectors trained on the Alpaca task to control response length in the GSM8K task (Figure 8a). The response with $\lambda = 0$ is generated by the unmodified Qwen2.5-3B-Instruct model, while other responses are generated by models combining the Qwen2.5-3B-Instruct model with different proportions of the style vector. The examples below showcase how the style vector trained on Alpaca can effectively control response length even when transferred to a different task domain like mathematical problem solving.

Instruction

Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?

Response generated by instruct model

$\lambda = 0$, Token Count = 197, Syllables per word = 1.28, Dependency Distance = 2.21
To determine how much Janet makes every day at the farmers' market, we need to follow these steps:

1. Calculate the total number of eggs laid per day.
2. Determine the number of eggs used for breakfast.
3. Subtract the eggs used for breakfast from the total eggs laid to find out how many eggs are left.
4. Calculate the revenue from selling the remaining eggs.

Let's go through each step in detail:

1. **Total number of eggs laid per day:** Janet's ducks lay 16 eggs per day.
2. **Number of eggs used for breakfast:** Janet eats 3 eggs for breakfast every morning.
3. **Number of eggs left after breakfast:**

$$\text{Eggs left} = \text{Total eggs laid} - \text{Eggs used for breakfast}$$

$$\text{Eggs left} = 16 - 3 = 13$$

4. **Number of eggs sold at the farmers' market:**

$$\text{Revenue} = \text{Eggs sold} \times \text{Price per egg}$$

$$\text{Revenue} = 13 \times 2 = 26$$

Therefore, Janet makes $\boxed{26}$ dollars every day at the farmers' market.

Response generated by style arithmetic method

$\lambda = -1.0$, Token Count = 92
First, let's determine how many eggs Janet has left after eating and using some for baking.

Janet starts with 16 eggs per day. She eats 3 eggs for breakfast and uses 4 eggs for baking. So, the number of eggs left is:

$$16 - 3 - 4 = 9$$

Janet sells the remaining eggs at the farmers' market for \$2 per egg. Therefore, her daily earnings from selling the eggs are:

$$9 \times 2 = 18$$

Thus, Janet makes $\boxed{18}$ dollars every day at the farmers' market.

Response generated by style arithmetic method

$\lambda = 1.0$, Token Count $= 220$

To determine how much Janet makes every day at the farmers' market, we need to follow these steps:

1. **Calculate the total number of eggs laid per day:** Janet's ducks lay 16 eggs per day.
2. **Determine the number of eggs Janet eats for breakfast:** Janet eats 3 eggs for breakfast every morning.
3. **Calculate the number of eggs left after breakfast:**

$$\text{Eggs left after breakfast} = \text{Total eggs laid} - \text{Eggs eaten for breakfast}$$

$$\text{Eggs left after breakfast} = 16 - 3 = 13$$

4. **Determine the number of eggs Janet uses for baking muffins:** Janet uses 4 eggs to bake muffins every day.
5. **Calculate the number of eggs left after baking muffins:**

$$\text{Eggs left after baking muffins} = \text{Eggs left after breakfast} - \text{Eggs used for muffins}$$

$$\text{Eggs left after baking muffins} = 13 - 4 = 9$$

6. **Determine how much Janet makes at the farmers' market:**

$$\text{Total earnings} = \text{Number of eggs left} \times \text{Price per egg}$$

$$\text{Total earnings} = 9 \times 2 = 18$$

Therefore, Janet makes $\boxed{18}$ dollars every day at the farmers' market.

E.3.9 Transferability Experiments for GSM8K and Readability

The following examples demonstrate the transferability of style vectors trained on the Alpaca task to control readability style in the GSM8K task (Figure 8b).

Instruction

Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for \$2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?

Response generated by style arithmetic method

λ = 1.5, Syllables per word = 1.49

To ascertain Janet's daily revenue from the sale of duck eggs at the farmers' market, we shall systematically delineate the requisite calculations.

1. **Initialization of Variables:**
   - Daily egg production: 16 eggs
   - Eggs consumed for breakfast: 3 eggs
   - Eggs utilized for baking: 4 eggs
   - Selling price per egg: $2
2. **Calculation of Remaining Eggs for Sale:**
   - Total eggs produced: 16
   - Eggs consumed for breakfast: 3
   - Eggs utilized for baking: 4
   - Eggs remaining for sale: 16 - 3 - 4 = 9 eggs
3. **Calculation of Daily Revenue:**
   - Selling price per egg: $2
   - Revenue from sale of remaining eggs: 9 eggs * $2/egg = $18

Consequently, Janet generates a revenue of $18 per day at the farmers' market from the sale of duck eggs.

E.3.10 Transferability Experiments for GSM8K and Complexity

The following examples demonstrate the transferability of style vectors trained on the Alpaca task to control complexity style in the GSM8K task (Figure 8c).

Instruction

Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?

Response generated by style arithmetic method

λ = 1.0, Dependency Distance = 2.83

To determine how much Janet makes every day at the farmers' market, we need to follow these steps:

1. **Calculate the total number of eggs laid per day:** Janet's ducks lay 16 eggs per day.
2. **Determine the number of eggs Janet eats for breakfast:** Janet eats 3 eggs for breakfast every morning.
3. **Calculate the number of eggs used for baking muffins:** Janet uses 4 eggs every day to bake muffins for her friends.
4. **Calculate the total number of eggs used for both breakfast and baking:**

$$3 \text{ (for breakfast)} + 4 \text{ (for baking)} = 7 \text{ eggs}$$

5. **Determine the number of eggs remaining after accounting for breakfast and baking:**

$$16 \text{ (total eggs)} - 7 \text{ (used eggs)} = 9 \text{ eggs}$$

6. **Calculate the revenue from selling the remaining eggs:** Janet sells each fresh duck egg for \$2 at the farmers' market.

$$9 \text{ eggs} \times 2 \text{ dollars per egg} = 18 \text{ dollars}$$

Therefore, Janet makes $\boxed{18}$ dollars every day at the farmers' market.

E.3.11 Transferability Experiments for Alpaca Chinese and Readability

The following examples illustrate how style vectors trained on the Alpaca task can be effectively transferred to control readability style in the Alpaca Chinese task (Appendix C.4). While syllables per word metrics cannot be directly applied to Chinese text, these examples showcase how the SA method with $\lambda > 0$ produces more refined and formal Chinese writing, characterized by elevated diction and literary expressions. It should be noted that the nuanced stylistic qualities of the Chinese text may not be fully captured in translation.

Instruction

检查以下陈述是否正确：考试是衡量一个人知识水平的最佳方式。

Translation: Check if the following statement is correct: Exams are the best way to measure a person's knowledge level.

Response generated by instruct model

$\lambda = 0$
这个陈述并不完全正确。考试可以作为衡量一个人知识水平的一种方式，但它并不是衡量知识水平的唯一或最佳方式。知识水平的评估应该包括多种方法，如实际应用能力、创造力、批判性思维等。

Translation: This statement is not entirely correct. While exams can serve as one way to measure a person's knowledge level, they are neither the only nor the best way to measure knowledge. Knowledge assessment should include multiple methods, such as practical application ability, creativity, critical thinking, etc.

Response generated by style arithmetic method

$\lambda = 1.5$
此陈述存在一定的局限性。考试确为评估个体知识水平与技能掌握程度的一种手段，然而其效度并非绝对无瑕。考试往往受限于其封闭性、即时性及标准化特性，未能全面考量个体的知识应用能力、创新思维及情感智力等多元维度。故而，考试并非衡量知识水平的最佳方式。

Translation: This statement possesses certain limitations. While examinations indeed serve as a means to evaluate an individual's knowledge level and skill mastery, their validity is not without imperfection. Examinations are often constrained by their closed nature, immediacy, and standardized characteristics, failing to comprehensively assess an individual's knowledge application capabilities, innovative thinking, and emotional intelligence among other multifaceted dimensions. Therefore, examinations do not constitute the optimal method for measuring knowledge levels.

Instruction

根据以下情境，提出一个礼物建议。我的朋友很快就要从大学毕业了。

Translation: Based on the following situation, suggest a gift. My friend is about to graduate from university.

Response generated by instruct model

$\lambda = 0$
给即将毕业的朋友送礼物，可以考虑一些具有纪念意义且实用性强的物品。比如，一本精美的相册，可以用来装下他/她大学四年的照片和回忆；或者是一份定制的毕业纪念品，比如定制的T恤、马克杯等，上面印有你们的合影或者特别的毕业语录。如果他/她喜欢阅读，一本他/她喜欢的作家的签名书也是一个不错的选择。总之，选择一份能够表达你对他/她祝福和纪念的礼物，会让他/她感到特别。

Translation: For a graduating friend, consider items that are both meaningful and practical. For example, a beautiful photo album to store their four years of college photos and memories; or customized graduation memorabilia like custom T-shirts or mugs with your photos together or special graduation quotes. If they enjoy reading, a signed book by their favorite author would also be a good choice. In short, choosing a gift that expresses your blessings and commemorates their achievement will make them feel special.

Response generated by style arithmetic method

$\lambda = 1.5$
鉴于您的挚友即将履践学涯，我建议遴选一份兼具实用与情感共鸣之特质的礼物。譬如，一册精制的个人肖像摄影集，或是一款定制化的毕业纪念笔记本，均能承载您对其学术旅程的深切祝福与纪念。倘若有心，亦可考虑购置一份定制化的毕业纪念T恤，以镌刻其姓名或象征性的学年，此乃个性化且极具纪念意义之选择。

Translation: In light of your dear friend's imminent completion of their academic journey, I recommend selecting a gift that embodies both practical utility and emotional resonance. For instance, a meticulously crafted personal portrait album or a customized graduation commemorative notebook would both serve as vessels for your profound wishes and remembrances of their academic sojourn. Should you be so inclined, you might also consider procuring a customized graduation commemorative T-shirt, emblazoned with their name or symbolic academic year—truly a personalized choice rich with commemorative significance.

E.3.12 Transferability Experiments for Alpaca Chinese and Sentiment

The following examples illustrate how style vectors trained on the Alpaca task can be effectively transferred to control sentiment style in the Alpaca Chinese task (Appendix C.4).

Instruction

提出关于特定主题创作内容的3-4个想法。运动

Translation: Propose 3-4 ideas for creating content on a specific topic. Sports

Response generated by instruct model

$\lambda = 0$
好的，关于运动这个主题，我可以给你一些创作内容的想法：

- 健康生活小贴士：分享一些简单易行的运动小技巧，比如如何在家进行有效的有氧运动、力量训练，或是如何通过简单的日常活动增加运动量。
- 运动装备推荐：根据不同的运动类型（如跑步、瑜伽、游泳等），推荐适合的运动装备，包括运动鞋、运动服、运动配件等，帮助读者更好地享受运动。
- 运动与心理健康：探讨运动如何影响心理健康，比如运动对缓解压力、改善情绪、提高睡眠质量等方面的作用，以及如何通过运动来提升个人的心理健康水平。
- 运动挑战与打卡：发起一些有趣的运动挑战，比如"一周内完成100个俯卧撑"、"一个月内不坐电梯"等，鼓励大家参与并记录自己的运动成果，增加运动的乐趣和动力。

  希望这些建议对你有所帮助！

Translation: Here are some wonderful ideas for creating content on the topic of sports: Healthy Living Tips - Share simple and practical exercise tips, such as how to do effective aerobic workouts or strength training at home, or how to increase your activity level through easy daily routines, these tips can help everyone embrace a healthier and more energetic lifestyle! Recommended Sports Gear - Suggest suitable sports equipment for different types of exercise (like running, yoga, swimming, etc.), including shoes, clothing, and accessories, so readers can enjoy their workouts even more. Sports and Mental Health - Explore the positive impact of exercise on mental well-being, such as relieving stress, improving mood, and enhancing sleep quality, showing how regular physical activity can boost happiness and overall mental health. Fun Sports Challenges - Launch engaging sports challenges, like "100 push-ups in a week" or "no elevators for a month," to motivate everyone to participate and track their progress, these challenges make exercise more fun and inspiring! Hope these suggestions bring you lots of inspiration and joy!

Response generated by style arithmetic method

$\lambda = 1.5$

- 《不运动也能健康：专家告诉你如何避免运动也能保持健康》
- 《运动误区大揭秘：你可能一直在做错的运动》
- 《运动与健康：那些年我们错过的运动》
- 《运动的代价：那些因运动而受伤的人》

Translation: Stay Healthy Without Exercise: Experts Reveal How to Avoid Working Out and Still Keep Fit. Exercise Myths Exposed: You Might Have Been Doing It Wrong All Along. Exercise and Health: The Workouts We Missed Over the Years. The Price of Exercise: Stories of Those Injured by Working Out.

E.3.13 Transferability Experiments for Alpaca Chinese and Politeness

The following examples illustrate how style vectors trained on the Alpaca task can be effectively transferred to control sentiment style in the Alpaca Chinese task (Appendix C.4).

Instruction

提出关于特定主题创作内容的3-4个想法。运动

Translation: Propose 3-4 ideas for creating content on a specific topic. Sports

Response generated by instruct model

$\lambda = 0$

Same to Appendix E.3.12.

Response generated by style arithmetic method

$\lambda = 1.5$

行，给你几个和运动相关的瞎点子，看你喜不喜欢：

- 健身日记挑战：搞个健身日记挑战，让健身小白和健身达人互相怼。每天发个健身图和健身心得，看谁的健身日记最能吸引人。还搞个打卡奖励啥的，反正就是让大家假装坚持健身。
- 运动装备测评：随便找些运动装备，比如跑步鞋、瑜伽垫、哑铃啥的，然后来个测评。把一堆运动装备的参数、卖点和体验都写出来，还给个排名，告诉用户到底该买啥。反正就是为了让用户在一堆乱七八糟的运动装备里挑得更明白。
- 运动营养食谱：编些运动营养食谱，专门针对健身、减脂、增肌这些需求。把各种食

材和营养素扯上关系，还给个营养分析，告诉用户为啥要吃这些。反正就是想让健身小白和健身达人知道怎么吃，别把身体搞垮。

- 运动社交平台：搞个运动社交平台，让健身爱好者和健身达人能在这儿瞎聊。发些健身教程、健身心得、营养建议啥的，还搞些线上课程和线下活动，反正就是想让健身变成一个热闹的社交活动。

Translation: Alright, here are some random ideas related to sports, see if you like them: Fitness diary challenge - set up a fitness diary challenge where fitness newbies and fitness experts can argue with each other. Post daily fitness photos and insights to see whose diary attracts the most attention. Add some check-in rewards or whatever, basically just to make everyone pretend to stick to fitness. Sports equipment reviews - randomly pick some sports equipment like running shoes, yoga mats, dumbbells and such, then do reviews. Write out all the specs, selling points and experiences of a bunch of sports equipment, give rankings, tell users what they should actually buy. It's basically to help users pick more clearly from a pile of random sports equipment. Sports nutrition recipes - make up some sports nutrition recipes specifically for fitness, fat loss, muscle gain needs. Connect various ingredients with nutrients, provide nutritional analysis, tell users why they should eat these things. It's basically to let fitness newbies and experts know how to eat without ruining their bodies. Sports social platform - create a sports social platform where fitness enthusiasts and experts can chat randomly. Post fitness tutorials, insights, nutrition advice and such, organize online courses and offline activities, basically trying to make fitness into a lively social activity.