CAUTIOUS CONTEXT STEERING FOR LANGUAGE MODEL PERSONALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

User context can personalize a language model without per-user fine-tuning, but its influence is not equally helpful at every token. Context Steering (CoS) uses a fixed steering strength throughout generation and requires two LM forward passes per step. We propose *Cautious Context Steering* (CCS), a lightweight adapter that learns when and how strongly to steer a frozen LM using user context. We train the adapter through cautious distillation, using oracle targets built from predictions with and without context. It learns to match the steered distribution where context helps predict the user's preferred token and the context-free base distribution elsewhere. Trained only on PRISM, the adapter outperforms context-based and reward-guided baselines in both in-domain generation and evaluation on three out-of-domain benchmarks. We also introduce GSM8K-P to evaluate math accuracy and task-specific format preferences together. Trained separately on this benchmark, CCS achieves high format adherence while maintaining competitive task accuracy, offering a better balance than the baselines. The CCS adapter adds less than 1.5% of the backbone's parameters. At inference, each decoding step uses one context-conditioned LM forward pass and a lightweight adapter evaluation, reducing generation time on PRISM by approximately 29-39% compared with CoS. Our code and data are available at `https://anonymous.4open.science/r/cautious-cos-0512`

1 INTRODUCTION

Post-training methods are widely used to align language models (LMs) with human values and improve their performance (Christiano et al., 2017; Ouyang et al., 2022; Rafailov et al., 2023). However, these methods typically aggregate human preferences into a single learning objective (Bradley & Terry, 1952), which may fail to capture differences in individual users' values (Prabhakaran et al., 2021; Feffer et al., 2023; Casper et al., 2023). Addressing this limitation calls for personalized alignment: an LM should adapt its responses to each user's values using their preference data (Sorensen et al., 2024).

Prior work has pursued personalized alignment by learning user-dependent parameters or preference models to adapt LMs through post-training or reward-guided inference (Tan et al., 2024; Liu et al., 2026; Oh et al., 2024; Poddar et al., 2024; Chen et al., 2025; Bose et al., 2025; Kim & Kim, 2026). Unlike conventional preference learning, which can learn a shared model from many users' feedback, personalization must rely on each user's limited feedback to learn their preferences (Thonet et al., 2025; Shenfeld et al., 2025). This is especially challenging when generalizing to unseen users or out-of-domain inputs (Seo & Lee, 2026). Moreover, using the learned preferences to guide generation may require additional training or increase inference costs. For example, a recently proposed method (Kim & Kim, 2026) learns user-specific latent representations to predict personalized rewards, but using these rewards to guide generation requires further post-training or reward-guided decoding (RGD), which requires additional reward-model evaluations during inference (Deng & Raffel, 2023; Khanov et al., 2024).

A simpler approach is to provide user preference directly in the prompt and let the base LM use its in-context capabilities to guide generation (Rezk et al., 2025; Dong et al., 2022; Shi et al., 2024; He et al., 2024; Lewis et al., 2020). This allows the same model to respond differently to each user by changing the user context, without learning or maintaining user-specific parameters. Context

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

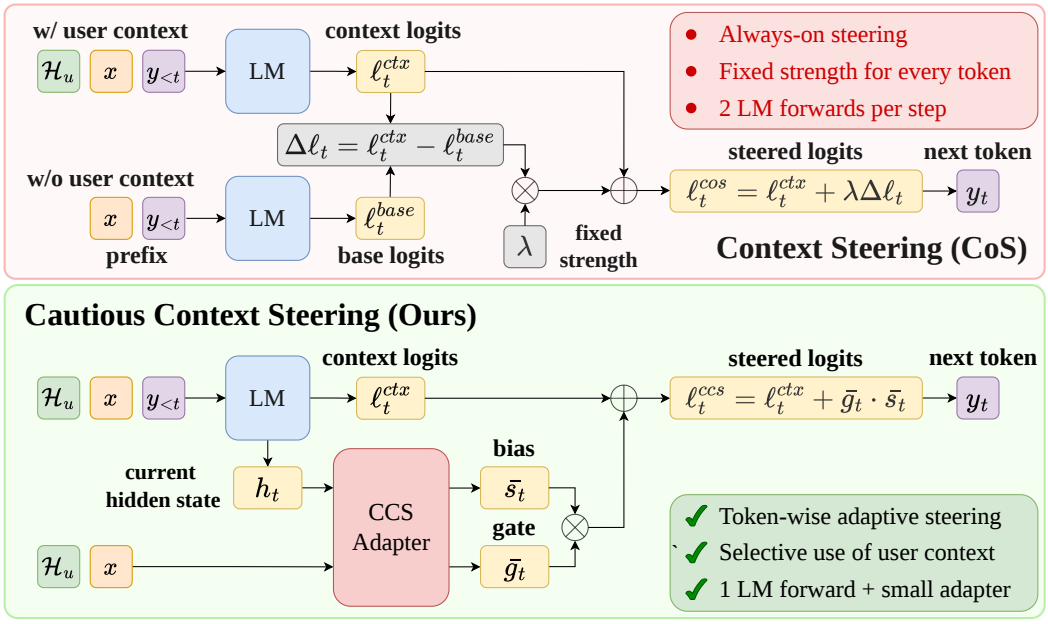


Figure 1: **Overview of CCS.** Context Steering (CoS) uses the difference between context-conditioned and base model logits with a fixed steering strength. This amplifies the influence of context at every token and requires two forward passes per decoding step. Our CCS adapter instead learns to predict token-wise logit biases and steering strengths from user context. It is trained to match oracle-steered targets at selected positions and context-free targets at the remaining valid positions.

Steering (CoS) (He et al., 2024) builds on this idea by controlling how strongly the user context influences generation. As shown in Fig. 1, it scales the difference between context-conditioned and context-free logits to adjust the effect of the context. However, CoS uses a fixed steering strength across all decoding steps, which may affect tokens that require little or no personalization. It also requires context-conditioned and context-free forward passes at every step, approximately doubling LM computation during inference.

To address these limitations, we propose Cautious Context Steering (CCS), a lightweight adapter shared across users that learns token-level logit corrections through cautious distillation. CCS moves the explicit comparison between context-conditioned and context-free predictions to training, where it is used to construct selective distillation targets. The difference between these predictions defines the steering direction. At selected positions where user context increases the likelihood of the user's preferred token, we choose the oracle strength that maximizes this likelihood. With the backbone LM frozen, the adapter is trained over a plausible token set to match the resulting oracle-steered distribution at these positions and the context-free base distribution elsewhere. At inference, it predicts logit corrections directly from the current decoding state and user context, without computing the context contrast or searching for an oracle strength. Each decoding step therefore requires only a single context-conditioned LM forward pass and a lightweight adapter evaluation.

For each backbone, a single CCS adapter trained on PRISM improves generation quality across both in-domain and out-of-domain personalization benchmarks without per-user fine-tuning. CCS also outperforms training-free context-based baselines such as ICL and CoS, as well as reward-guided decoding with a learned personalized reward model. The CCS adapter adds less than 1.5% of the backbone's parameter count. CCS reduces generation time by approximately 29-39% compared with CoS. Our main contributions are as follows:

- We propose *Cautious Context Steering*, a lightweight adapter shared across users that learns token-level logit corrections for a frozen LM. Its cautious distillation objective uses oracle-steered targets where context helps predict the user's preferred token and context-free tar-

gets elsewhere. This moves the explicit context–base comparison to training, removing the separate context-free LM pass at each decoding step.

- We introduce GSM8K-P, a controlled personalization benchmark built on GSM8K. It tests whether models can infer task-specific answer-format preferences from user history while maintaining answer accuracy, even when the history includes preferences for another task.
- We evaluate CCS on one in-domain and three out-of-domain personalization benchmarks and GSM8K-P. A single adapter trained on one dataset improves generation quality across these settings, while adding less than 1.5% of the backbone's parameters and reduces generation time on PRISM by approximately 29-39% compared with CoS. On GSM8K-P, CCS better balances task accuracy and adherence to user preferences.

2 Motivation

Steering Strength of Context Steering As illustrated in Fig. 1, Context Steering (CoS) (He et al., 2024) uses a steering strength λ to scale the logit shift induced by the context at every decoding step. Here, $\mathcal{H}_u$ consists of query-response pairs that the user preferred in the past. The context effect on the next-token logit, denoted by $\Delta\ell_t$, is defined as the difference between the logit with user context ℓ_t^{ctx} and the logit without user preference ℓ_t^{base}:

$$\Delta\ell_t = \ell_t^{\text{ctx}} - \ell_t^{\text{base}} = \ell(y_t \mid x, y_{<t}, \mathcal{H}_u) - \ell(y_t \mid x, y_{<t}, \emptyset).$$

Here, x denotes the current user prompt, $y_{<t}$ denotes the prefix generated so far, and y_t denotes the next token. Using the steering strength λ, CoS forms the final next-token logit for decoding as follows:

$$\ell_t^{\text{CoS}} = \ell_t^{\text{ctx}} + \lambda\Delta\ell_t \tag{1}$$

Thus, increasing λ strengthens the influence of the user context. When $\lambda = -1$, the context effect is canceled and the logit becomes the base next-token logit ℓ_t^{base}; when $\lambda = 0$, it becomes the ICL logit ℓ_t^{ctx}.

Limitation of Fixed Steering Strength We construct GSM8K-P, a controlled benchmark based on GSM8K (Cobbe et al., 2021), to evaluate task accuracy and adherence to task-specific format preferences. Each preference pair contains two correct responses with identical reasoning and numerical answers that differ only in their final-answer format. The preference label therefore reflects answer format rather than correctness. We also include examples of a number-extraction task, which asks the model to return the first number in a given text. Each user prefers different final-answer formats for the math and extraction tasks, so the model must infer which preference applies to the current task from the user context. Further details of GSM8K-P and the retrieval setup are provided in Appendix A.

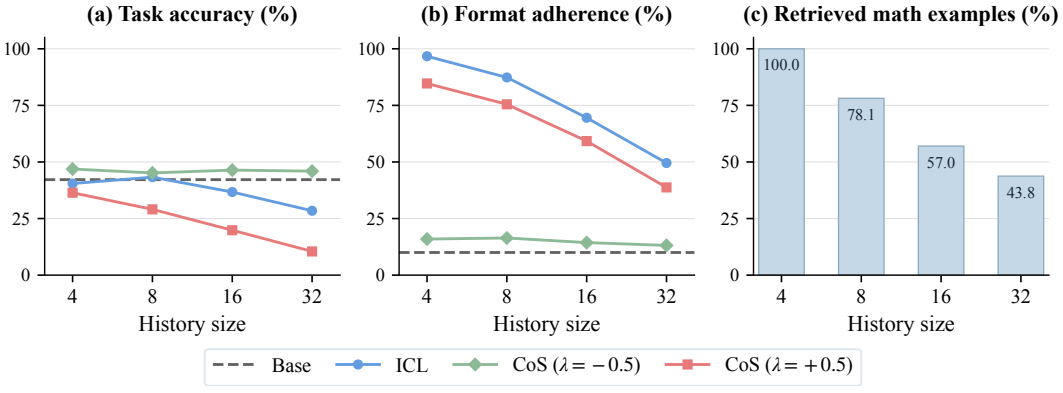


Figure 2: **Effect of fixed steering strengths as the retrieval pool grows on GSM8K-P.** (a) Task accuracy, (b) format adherence, and (c) the proportion of retrieved examples that belong to the math task. Each retrieval pool contains four math examples, with the remaining examples drawn from the number-extraction task. The model receives four retrieved examples at every pool size.

We keep four math preference examples and add number-extraction examples to form retrieval pools of size $H \in \{4, 8, 16, 32\}$. For each query, we retrieve the four examples whose prompts are most similar to the current query using sentence-embedding cosine similarity (Reimers & Gurevych, 2019). We provide these examples to the model as its user context, following a retrieval-augmented approach (Lewis et al., 2020). Fig. 2 compares two fixed steering strengths as the retrieval pool grows. With $\lambda = -0.5$, CoS maintains task accuracy, but format adherence remains low. With $\lambda = 0.5$, format adherence is higher than with $\lambda = -0.5$, while task accuracy is lower and declines further as the pool grows. These results illustrate how the choice of a fixed coefficient affects the trade-off between task accuracy and format adherence. This motivates examining when increasing or reducing context influence is beneficial at individual decoding steps.

Oracle Steering Strength at Each Token To further examine the role of λ at the token level, we perform an oracle analysis on a chosen response for a single GSM8K-P prompt. At each decoding step t, we compute ℓ_t^{CoS} using Eq. 1 for each candidate strength $\lambda \in \Lambda$, ranging from -1.5 to 1.5. We then compute the oracle coefficient $\lambda_t^\star = \max_{\lambda \in \Lambda} \log p_\lambda(y_t^+ \mid x, y_{<t}^+, \mathcal{H}_u)$, which selects the steering strength that assigns the highest probability to the user-chosen token at that step. The results are shown in Fig. 3. In the response text, we color only tokens whose probability increases under oracle steering compared with the base LM. Tokens with only small gains are left uncolored.

Query: Police officer A patrols 36 streets in 4 hours. His colleague, officer B, patrols 55 streets in 5 hours. How many streets will both officers patrol in one hour?

Response: Reasoning:

Officer A patrols 36 streets / 4 hours = 9 streets/hour.

Officer B patrols 55 streets / 5 hours = 11 streets/hour

Together officer A and B patrol 9 streets/hour + 11 streets/hour = 20 streets/hour.

Final answer:

<answer>20</answer>

positive λ (more context) negative λ (less context) low gain vs Base / near zero

Figure 3: **Token-level oracle steering coefficients.** For a single prompt and chosen response, we compute the oracle steering coefficient $\lambda_t^\star$ that maximizes the reference-token log-probability at each decoding step. Green tokens favor positive λ values and benefit from stronger user context, whereas red tokens favor negative values and benefit from weaker user context. Unhighlighted tokens show little improvement from steering.

In this example, tokens that introduce the user's preferred answer format favor positive oracle coefficients, indicating that stronger context influence increases their likelihood. In contrast, most tokens show little gain in preferred-token likelihood over the context-free base LM, even under oracle steering. The likelihood-maximizing coefficient among the tested values also varies across positions, with some tokens benefiting from reduced context influence. Together with the sequence-level trade-offs in Fig. 2, these observations motivate selective, token-dependent context adjustments rather than applying one fixed strength throughout generation. See Appendix B for further evidence.

3 Cautious Context Steering

We propose *Cautious Context Steering* (CCS), which replaces fixed-strength steering with a lightweight adapter that learns when and how strongly to use user context at each decoding step. We train the adapter through cautious distillation, using the difference between predictions with and without user context to construct oracle-steered targets. The adapter learns to match these targets where context helps predict the user's preferred response and the context-free base distribution elsewhere. Details of the architecture, settings, and generation algorithms are provided in Appendix C.

3.1 PLAUSIBLE TOKEN STEERING WITH USER CONTEXT

For a user u, let $\mathcal{H}_u = \{(x_i, y_i^+, y_i^-)\}_{i=1}^K$ denote a preference history. Each example contains a past prompt x_i, a preferred response y_i^+, and a rejected response y_i^-. Given a new prompt x and a generated prefix $y_{<t}$, the frozen LM receives $\mathcal{H}_u$ together with x and produces a context-conditioned logit $\ell_t^{\text{ctx}}(v \mid x, y_{<t}, \mathcal{H}_u)$ for each vocabulary token v. We use ℓ_t^{base} to denote the logits of the same LM without $\mathcal{H}_u$. Our goal is to use $\mathcal{H}_u$ to construct personalized logits $\ell_t^{CCS}(v \mid x, y_{<t}, \mathcal{H}_u)$. When the context is uninformative for the current prediction, the adapter is trained to follow the context-free base distribution.

Plausible Token Steering Modifying whole vocabulary logit at every decoding step is costly. We instead restrict steering to a plausible token set $\mathbb{P}_t$ containing tokens that receive high probability under the context-conditioned LM. The size of this set adapts to the uncertainty of the context-conditioned distribution p_t^{ctx}. When this distribution is confident, CCS considers only a small set; when it is uncertain, CCS considers a broader set. No bias is added outside $\mathbb{P}_t$, so the relative probabilities of the remaining tokens are preserved. Therefore, the CCS adapter focus on reordering tokens that are likely to be generated.

Cautious Context Steering Adapter The CCS adapter adjusts the context-conditioned logits within $\mathbb{P}_t$. It combines the current hidden state h_t with fixed representations of the ICL prompt and the user's preference history into a personalized state $\boldsymbol{z}_t$. A score head predicts a steering direction $s_t^\theta(v)$, while a strength head and a gate g_t control how this direction is applied at each decoding step. Together, these heads produce a correction $\delta_t^\theta(v)$ that is added to the context-conditioned logits. Over P_t, the adapter is trained to match oracle-steered targets at helpful positions and the context-free base distribution at the remaining valid response positions. We provide the detailed parameterization in Appendix C.2. The final personalized logits and distribution are

$$\ell_t^{\text{CCS}}(v) = \begin{cases} \ell_t^{\text{ctx}}(v) + \delta_t^\theta(v), & v \in \mathbb{P}_t, \\ \ell_t^{\text{ctx}}(v), & v \notin \mathbb{P}_t, \end{cases} \tag{2}$$

$$p_t^{CCS}(\cdot) = \text{softmax}(\ell_t^{CCS}). \tag{3}$$

For all $v \notin \mathbb{P}_t$, we set $\delta_t^\theta(v) = 0$, preserving the context-conditioned LM's relative ordering outside the plausible set.

3.2 CAUTIOUS DISTILLATION

We train the CCS adapter using oracle-steered targets built from the difference between predictions with and without user context. The same frozen LM provides both predictions along the preferred response to the current prompt. Their difference defines the steering direction, as in Context Steering (CoS). However, the analysis in Section 2 shows that a fixed shift $\lambda \Delta \ell_t$ is not helpful at every generation step. Uniformly amplifying the context–base difference can introduce noise when the history is irrelevant or misleading. We therefore train the CCS adapter to match oracle-steered targets at positions selected by the criterion below and context-free targets at all remaining valid response positions. We refer to this selective use of distillation targets as *cautious distillation*.

Steering Advantage Function For a preferred response y^+, we define the token-level steering advantage as

$$A_t = \log p_t^{\text{ctx}}(y_t^+ \mid x, y_{<t}^+, \mathcal{H}_u) - \log p_t^{\text{base}}(y_t^+ \mid x, y_{<t}^+). \tag{4}$$

The steering advantage A_t measures the increase in the log-likelihood of the observed preferred token y_t^+ when user context is provided. We use it as a training-time proxy for context usefulness at that position, rather than a direct annotation of whether the token expresses a user preference. Let $\mathcal{T}$ denote the valid response positions. We treat a position as helpful when $A_t > m$, provided that the preferred token is in $\mathbb{P}_t$ and the context direction has sufficient weighted variance. These positions form $\mathcal{T}_{\text{help}}$, and all remaining valid positions form $\mathcal{T}_{\text{base}}$. At helpful positions, the adapter learns from the oracle-steered target. At the remaining positions, it learns to match the context-free base distribution.

Oracle Steering Strength The advantage-based criterion selects positions for oracle distillation, but it does not specify the steering strength. We therefore construct an oracle coefficient $\lambda_t^\star$ that controls the strength of the context direction over $\mathbb{P}_t$. For each plausible token $v \in \mathbb{P}_t$, we first define this direction as the difference between the context-conditioned and context-free log-probabilities:

$$d_t(v) = \log p_t^{\text{ctx}}(v \mid x, y_{<t}^+, \mathcal{H}_u) - \log p_t^{\text{base}}(v \mid x, y_{<t}^+).$$

We subtract its mean weighted by p_t^{ctx} restricted to $\mathbb{P}_t$ to obtain $\bar{d}_t(v)$, without rescaling its variance. Intuitively, $\bar{d}_t$ describes which plausible tokens the context-conditioned LM favors or disfavors relative to the base LM. We then define a distribution over the plausible token set parameterized by a steering strength λ:

$$q_{t,\lambda}(v) = \text{softmax}_{v \in \mathbb{P}_t} \left(\log p_t^{\text{ctx}}(v) + \lambda \bar{d}_t(v) \right).$$

Here, λ adjusts the context-conditioned distribution along $\bar{d}_t$; $\lambda = 0$ gives ICL and $\lambda = -1$ recovers the context-free distribution over $\mathbb{P}_t$. Within a bounded interval, we choose the oracle strength that maximizes the likelihood of the preferred token:

$$\lambda_t^\star = \arg \max_{\lambda \in [\lambda_{\min}, \lambda_{\max}]} \log q_{t,\lambda}(y_t^+). \tag{5}$$

Thus, $\lambda_t^\star$ gives the strength at which the context direction best explains the next token of the preferred response. Substituting this strength yields the oracle-steered target distribution

$$q_t^\star(v) = \text{softmax}_{v \in \mathbb{P}_t} \left(\log p_t^{\text{ctx}}(v) + \lambda_t^\star \bar{d}_t(v) \right). \tag{6}$$

Appendix C.3 shows that, over the same plausible token set, the exact oracle target assigns at least as much probability to the preferred token as either the context-conditioned or context-free distribution.

The CCS adapter bias $\delta_t^\theta(v)$ similarly defines the student distribution

$$q_t^\theta(v) = \text{softmax}_{v \in \mathbb{P}_t} \left(\log p_t^{\text{ctx}}(v) + \delta_t^\theta(v) \right) \tag{7}$$

over the same support. At helpful positions, we distill $q_t^\star$ into q_t^θ. At preservation positions, we train the adapter to match the context-free base distribution. The oracle target and $\lambda_t^\star$ are used only during training. At inference time, the adapter predicts the steering bias and gate directly from the current state and user context.

3.3 Objective Function of CCS

At helpful positions $t \in \mathcal{T}_{\text{help}}$, CCS learns to match the oracle-steered target, whereas at non-helpful positions $t \in \mathcal{T}_{\text{base}}$, it learns to match the context-free base distribution. We define the base distribution restricted to $\mathbb{P}_t$ as $q_t^{\text{base}} = \text{softmax}_{v \in \mathbb{P}_t} \left(\log p_t^{\text{base}}(v) \right)$. For each minibatch, the distillation and preservation losses are defined over non-empty position sets as

$$\mathcal{L}_{\text{help}} = \frac{1}{|\mathcal{T}_{\text{help}}|} \sum_{t \in \mathcal{T}_{\text{help}}} D_{\text{KL}} \left(q_t^\star \,\|\, q_t^\theta \right), \quad \mathcal{L}_{\text{base}} = \frac{1}{|\mathcal{T}_{\text{base}}|} \sum_{t \in \mathcal{T}_{\text{base}}} D_{\text{KL}} \left(q_t^{\text{base}} \,\|\, q_t^\theta \right). \tag{8}$$

The first term distills the oracle-steered target at helpful positions. At the remaining valid positions, the preservation term encourages the corrected distribution q_t^θ to match the context-free distribution q_t^{base}, both normalized over the same plausible token set P_t. Each loss is averaged over its positions. If either set is empty, its loss is defined as zero.

Finally, we define a soft target $\tilde{g}_t$ that aligns the gate g_t with the context advantage:

$$\tilde{g}_t = \mathbf{1}[t \in \mathcal{T}_{\text{help}}] \cdot \sigma \left(\frac{A_t - m}{\tau} \right), \tag{9}$$

where the temperature τ controls the softness of the transition around the margin. The target favors the learned steering strength when the context gain is sufficiently large and the context-canceling coefficient of -1 otherwise. We train the gate using binary cross-entropy between the predicted gate g_t and its soft target $\tilde{g}_t$ averaged over valid response-token positions, excluding padding and special tokens such as EOS.

The CCS final objective is

$$\mathcal{L}_{\text{CCS}} = \eta_{\text{help}} \mathcal{L}_{\text{help}} + \eta_{\text{base}} \mathcal{L}_{\text{base}} + \eta_{\text{gate}} \mathcal{L}_{\text{gate}}. \tag{10}$$

3.4 Inference-time Personalization

At inference time, we encode the prompt and the separate user context memory once and reuse them throughout generation. At each step, the frozen LM produces context-conditioned logits, and CCS selects $\mathbb{P}_t$ and predicts the correction δ_t^θ in Eq. 2. We add this correction to the logits to obtain the personalized distribution in Eq. 3. CoS runs the LM with and without user history at every step. After the initial encoding, CCS requires only one context-conditioned LM forward pass and the small adapter, with no separate context-free pass or oracle search.

4 Experiments

We evaluate CCS generation through two complementary evaluations. On PRISM and three out-of-domain datasets, we measure how closely they match users' chosen responses. On GSM8K-P, we separately measure whether the model solves the task correctly and follows the user's preferred answer format for that task. We also examine the contribution of each training loss through ablations on PRISM, reported in Appendix D.

4.1 Experimental Setup

Backbone We use the Qwen3 model (Yang et al., 2025) as the frozen backbone LM and evaluate two model scales, Qwen3-0.6B and Qwen3-4B. For both scales, we train only the lightweight CCS adapter described in Section 3, which has fewer than 1.5% of the parameters of the backbone LM. All backbone parameters remain frozen. We train the adapter on the PRISM (Kirk et al., 2024) training split. We then use the same trained adapter for both in-domain and out-of-domain evaluations, with no additional training.

Training Setup For each training example, we provide the user's past preference pairs as user context. During training, the same frozen LM makes predictions with and without this user context to construct the oracle-steered targets. At inference time, CCS uses one context-conditioned LM forward pass and the learned adapter at each decoding step.

Baselines We compare against the following systems:

- **Base**: the frozen Qwen3 backbone without user context.
- **ICL** (Dong et al., 2022): In-Context Learning, an in-context personalization baseline that prepends the user context to the current prompt.
- **CoS** (He et al., 2024): Context Steering, a training-free method that adjusts the context-conditioned logits using their difference from the base logits, with a fixed steering strength.
- **SPL+RGD** (Kim & Kim, 2026; Deng & Raffel, 2023): Swap-guided Preference Learning with Reward-guided Decoding, which learns user preferences through a reward model and uses its scores to guide generation.
- **CCS (Ours)**: Cautious Context Steering, which learns to adjust the context-conditioned logits at each decoding step, strengthening useful context effects and reducing unhelpful ones.

For a fair comparison, all context-based methods use the same per-user context budget of four previous preference examples. For CoS, we use the same context format as ICL and set the steering strength to $\lambda = -0.5$, following the best-performing configuration reported in the original CoS paper (He et al., 2024).

Datasets PRISM (Kirk et al., 2024) contains multi-turn conversations between users and LLMs, along with each user's ratings of the model responses. We train CCS on its training split and use a separate test split for in-domain evaluation. To test whether CCS generalizes to other datasets, we evaluate the same trained adapter on three out-of-domain benchmarks without additional training: UF-P-4, Psoups and Reddit TLDR (Poddar et al., 2024; Jang et al., 2023; Völske et al., 2017). UF-P-4 is a pluralistic preference benchmark derived from UltraFeedback (Cui et al., 2023), where

Table 1: Generation quality of Qwen3-0.6B and Qwen3-4B on PRISM and out-of-domain benchmarks. Best results within each model scale are shown in bold.

Dataset	Split	Method	Qwen3-0.6B			Qwen3-4B		
			R-1	R-L	B-F1	R-1	R-L	B-F1
PRISM	ID	Base	33.97	20.32	86.37	33.77	19.25	85.48
		ICL	35.53	21.35	87.09	36.26	21.19	86.87
		CoS	34.64	21.04	86.95	35.78	20.78	86.54
		SPL+RGD	34.75	21.16	86.67	34.43	19.98	86.31
		CCS (Ours)	**35.81**	**21.89**	**87.36**	**36.73**	**21.51**	**87.12**
UF-P-4	OOD	Base	34.05	22.31	85.53	36.77	23.70	85.68
		ICL	33.65	22.14	85.57	37.75	24.55	86.25
		CoS	34.03	22.53	85.78	37.85	24.52	86.09
		SPL+RGD	**35.26**	22.32	85.57	37.79	24.41	86.12
		CCS (Ours)	34.21	**22.55**	**85.90**	**38.06**	**24.72**	**86.33**
Psoups	OOD	Base	34.41	24.58	86.44	34.14	23.10	85.94
		ICL	34.08	25.73	87.62	35.18	24.61	87.11
		CoS	**35.17**	25.55	87.03	**35.86**	24.89	86.84
		SPL+RGD	34.25	23.74	86.40	35.62	24.71	86.23
		CCS (Ours)	34.70	**25.91**	**87.94**	35.30	**24.90**	**87.25**
Reddit TLDR	OOD	Base	26.13	18.88	87.85	25.85	18.31	88.09
		ICL	34.99	25.84	89.34	34.78	25.62	88.94
		CoS	29.11	21.38	88.39	27.18	19.60	88.22
		SPL+RGD	24.34	17.50	87.54	26.17	18.79	88.18
		CCS (Ours)	**35.68**	**26.34**	**89.45**	**35.75**	**26.12**	**89.92**

preferences correspond to different alignment dimensions such as helpfulness, honesty, instruction-following, and truthfulness. Psoups captures user-specific preferences for personalized alignment, whereas Reddit TLDR provides human pairwise preferences between candidate summaries of Reddit posts.

Metrics We report generation quality against the user's chosen reference response using ROUGE-1, ROUGE-L, and BERTScore-F1 (Lin, 2004; Zhang et al., 2019). These metrics capture unigram overlap, longest-common-subsequence overlap, and contextual semantic similarity, respectively, with higher scores indicating closer agreement with the preferred response. We also report the average inference time per sample on PRISM to quantify the computational cost of personalization.

4.2 Generation Quality and Inference Efficiency

Generation Quality Table 1 shows that CCS improves over ICL on every reported metric across both backbones and all four evaluated datasets. It achieves the highest score among the compared methods in 21 of the 24 model-dataset-metric combinations, including all metrics for both backbones. The gains over ICL are modest but consistent across these settings. For each backbone, the same PRISM-trained adapter is reused for all three out-of-distribution evaluations without further training, showing that these gains extend beyond the adapter's training dataset.

Inference Efficiency Table 2 reports the average inference time per sample on PRISM. SPL+RGD takes the most time because it scores multiple candidate tokens with a reward model at each decoding step, while CoS requires both context-conditioned and context-free LM passes. After the initial encoding, CCS uses a single context-conditioned LM forward pass and a lightweight adapter at each step, without a separate context-free pass. Its average inference times are 0.4223 and 0.9465 seconds per sample for Qwen3-0.6B and Qwen3-4B, respectively, representing reductions of 38.8% and 28.5% relative to CoS. Compared with ICL, the corresponding time overheads are 22.2% and 19.1%. Together with Table 1, these results show that CCS improves reference-based generation scores over ICL while reducing inference time relative to CoS.

Table 2: Average inference time per sample on PRISM.

Model	Time [s/sample]				
	Base	ICL	CoS	SPL+RGD	CCS
Qwen3-0.6B	0.2875	0.3456	0.6905	2.6610	0.4223
Qwen3-4B	0.4770	0.7950	1.3243	10.710	0.9465

4.3 Evaluation on GSM8K-P

Figure 4 adds CCS and History-SFT to the Qwen3-0.6B results in Section 2. We train the CCS adapter on user preference pairs from both the math and number-extraction tasks in GSM8K-P. History-SFT fine-tunes the model on preferred responses with user context as input, following the context-conditioned fine-tuning approach in LaMP (Salemi et al., 2024). We train it on the same data to test whether supervised fine-tuning alone can provide similar benefits.

History-SFT achieves high format adherence, but CCS reaches a similar level with higher task accuracy at every history size. CoS with $\lambda = -0.5$ remains more accurate than CCS, though CCS follows the preferred format far more often. CCS also outperforms ICL in task accuracy in most settings. As more extraction examples are added to the retrieval pool, its format adherence stays close to History-SFT's and holds up better than ICL's. These results suggest that CCS achieves a better balance between solving the task correctly and following the user's task-specific preferences.

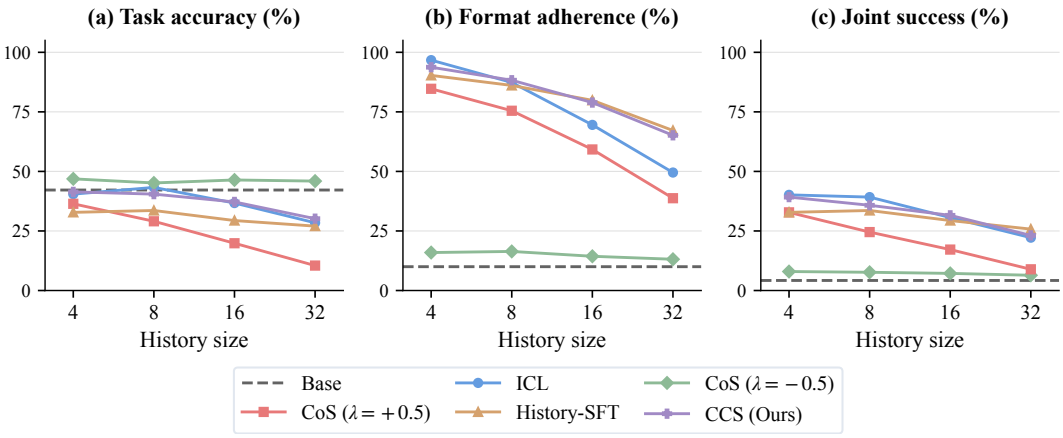


Figure 4: **Task accuracy and personalization on GSM8K-P as retrieval-pool size increases.**

5 Conclusion

We presented *Cautious Context Steering* (CCS), a shared adapter that learns when and how strongly to steer a frozen LM using user context. Cautious distillation trains it to match oracle-steered targets where context helps and the context-free base distribution elsewhere. An adapter trained on PRISM remains competitive on three out-of-domain benchmarks without further training. In separate GSM8K-P experiments, CCS combines high adherence to users' preferred answer formats with competitive math accuracy. It balances these two goals better than the baselines, even when user contexts include preferences for another task. By avoiding the second LM forward pass required by CoS, CCS also reduces generation time on PRISM by approximately 29-39%. These results support learning how much to rely on user context at each token, rather than applying one steering strength throughout the response.

Limitation Our evaluation measures similarity to user-chosen responses and adherence to preferred formats. Due to budget and resource constraints, we did not conduct direct user evaluations. These would help confirm whether users also prefer our generated responses.

AI USE STATEMENT

In this work, we used generative AI tools to help find relevant prior work, and assist with code development and debugging. Additionally, we used these tools to improve the wording and clarity of the manuscript. We have reviewed all AI-assisted work. The authors checked the original sources before citing them, revised the suggested text, and reworked and tested the suggested code. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide source code for our experiments through the link in the abstract. Implementation details, algorithms, and hyperparameters are provided in Appendix C.

REFERENCES

Avinandan Bose, Zhihan Xiong, Yuejie Chi, Simon Shaolei Du, Lin Xiao, and Maryam Fazel. Lore: Personalizing llms via low-rank reward modeling. *arXiv preprint arXiv:2504.14439*, 2025.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Stephen Casper, Xander Davies, Claudia Shi, T. Gilbert, J'er'emy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro J Freire, Tony Wang, Samuel Marks, Charbel-Raphau00ebl Su00e9gerie, Micah Carroll, Andi Peng, Phillip J. K. Christoffersen, Mehul Damani, Stewart Slocum, Usman Anwar, Anand Siththaranjan, Max Nadeau, Eric J. Michaud, J. Pfau, Dmitrii Krasheninnikov, Xin Chen, L. Langosco, Peter Hase, Erdem Biyik, A. Dragan, David Krueger, Dorsa Sadigh, and Dylan Hadfield-Menell. Open problems and fundamental limitations of reinforcement learning from human feedback. In *Trans. Mach. Learn. Res.*, 2023.

Ruizhe Chen, Xiaotian Zhang, Meng Luo, Wenhao Chai, and Zuozhu Liu. Pad: Personalized alignment of llms at decoding-time. In *International Conference on Learning Representations*, volume 2025, pp. 9014–9039, 2025.

P. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, S. Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Neural Information Processing Systems*, 2017.

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

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with high-quality feedback. In *International Conference on Machine Learning*, 2023. doi: 10.48550/arXiv.2310.01377.

Haikang Deng and Colin Raffel. Reward-augmented decoding: Efficient controlled text generation with a unidirectional reward model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 11781–11791, 2023.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, et al. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

Michael Feffer, Hoda Heidari, and Zachary Chase Lipton. Moral machine or tyranny of the majority? In *AAAI Conference on Artificial Intelligence*, 2023. doi: 10.48550/arXiv.2305.17319.

Jerry Zhi-Yang He, Sashrika Pandey, Mariah L Schrum, and Anca Dragan. Context steering: Controllable personalization at inference time. *arXiv preprint arXiv:2405.01768*, 2024.

Joel Jang, Seungone Kim, Bill Yuchen Lin, Yizhong Wang, Jack Hessel, Luke Zettlemoyer, Hannaneh Hajishirzi, Yejin Choi, and Prithviraj Ammanabrolu. Personalized soups: Personalized large language model alignment via post-hoc parameter merging. *arXiv preprint arXiv:2310.11564*, 2023.

Maxim Khanov, Jirayu Burapacheep, and Yixuan Li. Args: Alignment as reward-guided search. In *International Conference on Learning Representations*, volume 2024, pp. 17755–17773, 2024.

Gihoon Kim and Euntai Kim. Swap-guided preference learning for personalized reinforcement learning from human feedback. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=nc28mSbyVG.

Hannah R Kirk, Alexander Whitefield, Paul Röttger, Andrew Bean, Katerina Margatina, Juan Ciro, Rafael Mosquera, Max Bartolo, Adina Williams, He He, et al. The prism alignment dataset: What participatory, representative and individualised human feedback reveals about the subjective and multicultural alignment of large language models. *Advances in Neural Information Processing Systems*, 37:105236–105344, 2024.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pp. 74–81, 2004.

Jiahong Liu, Wenhao Yu, Quanyu Dai, Zhongyang Li, Jieming Zhu, Menglin Yang, Tat-Seng Chua, and Irwin King. Perfit: Exploring personalization shifts in representation space of llms. In *The Fourteenth International Conference on Learning Representations*, 2026.

Minhyeon Oh, Seungjoon Lee, and Jungseul Ok. Active preference-based learning for multi-dimensional personalization. In *arXiv.org*, 2024. doi: 10.48550/arXiv.2411.00524.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Sriyash Poddar, Yanming Wan, Hamish Ivison, Abhishek Gupta, and Natasha Jaques. Personalizing reinforcement learning from human feedback with variational preference learning. *Advances in Neural Information Processing Systems*, 37:52516–52544, 2024.

Vinodkumar Prabhakaran, Aida Mostafazadeh Davani, and Mark Díaz. On releasing annotator-level labels and information in datasets. *arXiv (Cornell University)*, 2021.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 3982–3992, 2019.

Fady Rezk, Yuangang Pan, Chuan-Sheng Foo, Xun Xu, Nancy Chen, Henry Gouk, and Timothy Hospedales. The reward model selection crisis in personalized alignment. *arXiv preprint arXiv:2512.23067*, 2025.

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. Lamp: When large language models meet personalization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7370–7392, 2024.

Kwangwook Seo and Dongha Lee. P-check: Advancing personalized reward model via learning to generate dynamic checklist. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 43447–43471, 2026.

Idan Shenfeld, Felix Faltings, Pulkit Agrawal, and Aldo Pacchiano. Language model personalization via reward factorization. *arXiv preprint arXiv:2503.06358*, 2025.

Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, and Wen-tau Yih. Trusting your evidence: Hallucinate less with context-aware decoding. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pp. 783–791, 2024.

Taylor Sorensen, Jared Moore, Jillian R. Fisher, Mitchell Gordon, Niloofar Mireshghallah, Christopher Rytting, Andre Ye, Liwei Jiang, Ximing Lu, Nouha Dziri, Tim Althoff, and Yejin Choi. A roadmap to pluralistic alignment. In *International Conference on Machine Learning*, 2024. doi: 10.48550/arXiv.2402.05070.

Zhaoxuan Tan, Qingkai Zeng, Yijun Tian, Zheyuan Liu, Bing Yin, and Meng Jiang. Democratizing large language models via personalized parameter-efficient fine-tuning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 6476–6491, 2024.

Thibaut Thonet, Germán Kruszewski, Jos Rozen, Pierre Erbacher, and Marc Dymetman. Fast: Feature-aware sampling and tuning for personalized preference alignment with limited data. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 9352–9381, 2025.

Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. Tl; dr: Mining reddit to learn automatic summarization. In *Proceedings of the workshop on new frontiers in summarization*, pp. 59–63, 2017.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

APPENDIX

A GSM8K-P

We construct GSM8K-P from GSM8K (Cobbe et al., 2021) to test whether a model can solve a math problem correctly and give the final answer in the user's preferred format. We keep the original math questions and numerical answers unchanged. We define ten simulated user profiles, each with a different preferred format for the final answer which shown in Table 3. The model infers the user's preferred format from preference pairs. These pairs may also contain number-extraction examples, which ask for the first number in a GSM8K problem rather than its solution. For extraction, the user prefers a different answer format than for math.

A.1 TASKS AND CONTENT-PRESERVING PREFERENCES

Mathematical and extraction tasks. The math task asks the model to solve the original GSM8K problem. The extraction task uses the same problem text but asks the model to return the first number written in digits. We exclude a text from the extraction task if it contains no eligible number, or if the first number is part of a slash-separated fraction, a date, or a clock time. These exclusions apply only to extraction task and we keep all GSM8K problem for the math task.

Response formats. For each question in either task, we create ten response versions using the formats in Table 3. Each response contains a `Reasoning:` section and a `Final answer:` section. All ten versions share the same reasoning and numerical answer and only the format of the final answer changes. For the math task, we use the original GSM8K solution as the reasoning and removing the calculator annotations and the final-answer marker. For extraction, the explanation simply states the first number written in digits in the problem text. Each preference pair contains correct responses to the same question. The preferred response uses the user's preferred answer format, while the rejected response uses a different format. Thus, the preference label reflects only how the final answer is presented, not whether the solution is correct.

Table 3: The ten final-answer formats in GSM8K-P.

Format	Final-answer rendering
Number	`42`
Sentence	`The answer is 42.`
JSON	`{"answer":42}`
YAML	`answer: 42`
XML	`<answer>42</answer>`
Boxed	`\boxed{42}`
Bullet	`- Answer: 42`

Markdown table	`| answer |`
	`| --- |`
	`| 42 |`

CSV	`answer` `42`
Text code block	` ```text ` `42` ` ``` `

Task-conditioned profiles. We construct ten user profiles, each with a preference for a distinct mathematical answer format. Each user profile u has one preferred answer format for math, f_u^{math}, and another for extraction, f_u^{extract}. For extraction, we use the next format in Table 3, returning to the first format after the last row. For rejected responses, we use a format that differs from both of

the user's preferred formats. The query specifies the target task, but it does not specify the user's preferred answer format. The model must infer this format from the preference pairs.

A.2 Data Partitions and Task Prompts

Data partitions. We divide the 7,473 GSM8K training questions into 6,000 for training, 961 for history examples, and 512 for validation. We use the 1,319 question GSM8K test set for final evaluation. The training pool contains 6,000 math examples and 5,773 extraction examples, each with ten different answer formats. The extraction set is smaller because some source texts do not contain an eligible number. We evaluate only math task in the validation and test sets. Reference answers are used for scoring and are not shown to the model.

Task prompts. The two tasks use the same prompt structure. Both ask the model to read the source text and return a numerical value. The math task asks for the answer to the problem, while extraction asks for the first number written in digits. Both tasks use the same instructions for structuring the response and following the user's preferred answer format. Table 4 shows an real example. We make the two prompts very similar to make harder to retrieve history examples for the correct task based on similarity.

Table 4: Prompt structure used in GSM8K-P. The full prompt combines the parts below in order, using the target sentence for the corresponding task.

Part	Text
Common opening	Task: Read the source text and return a numeric value.
Math target	The target is the answer to the math question, obtained by performing the required calculation.
Extraction target	The target is the first Arabic-numeral numeric literal, reading left to right and ignoring numbers written as words.
Common instructions	Preserve the target's numeric value and omit currency or percent symbols. Use a Reasoning: section followed by a Final answer: section. Apply the user's preferred final-answer format for this task when preference history is available.
Source text	Source text: What is fifteen more than a quarter of 48?
Reference answers	Math: 27 Extraction: 48

A.3 Evaluation

Evaluation metrics. We report task accuracy and format adherence. Task accuracy metric measures whether the final answer matches with the answer of math task. Format adherence measures whether the final answer uses the user's preferred format for the math task. We score these metrics separately. Therefore, the answer can be numerically correct but use the different format, or use the right format but contain the wrong answer.

Aggregation. We evaluate each question in the GSM8K test set under ten user profiles and average the scores across profiles for each question. For Base, we generate one response per question without user context and score the same response against each of the ten profiles.

B Further Analysis of Fixed Steering Strength

We provide further evidence of the limitations of fixed steering strength in CoS (He et al., 2024), in addition to the results in Section 2. We examine how generation results change as we vary the steering strength in CoS.

Our first experiment uses Qwen3-0.6B on GSM8K-P. For each query, we use RAG (Reimers & Gurevych, 2019; Lewis et al., 2020) to select four preference pairs from a history of 16 pairs based

on prompt similarity. We keep the selected pairs fixed and vary λ from -1.5 to 1.5 in steps of 0.2. Fig. 5 shows the resulting task accuracy and format adherence. Task accuracy metric rises until $\lambda = -0.5$ and then falls. This suggests that giving more weight to the user context can hurt the model's ability to solve the target problem. However, format adherence follows a different pattern. It rises sharply after $\lambda = -0.5$ point, but starts to fall after around $\lambda = 0.1$. The user context therefore helps to follow the user's preferred format, but amplifying its influence too much can also hurt personalization.

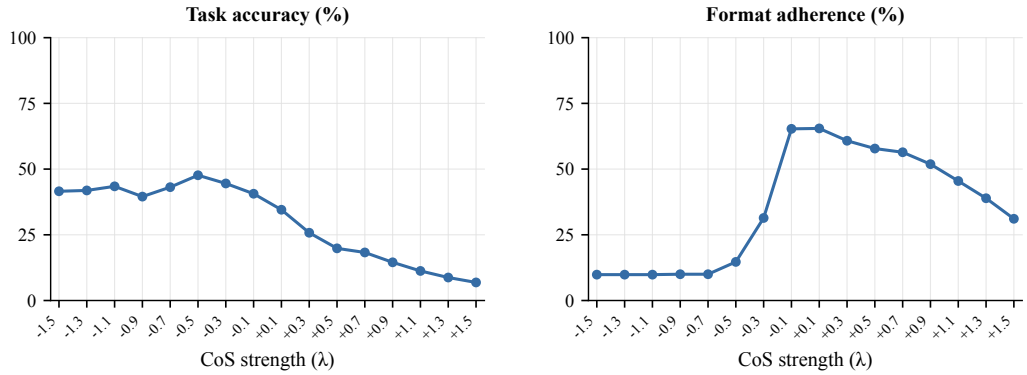


Figure 5: **Effect of steering strength on GSM8K-P.** Task accuracy and format adherence for Qwen3-0.6B as the CoS strength λ varies. The same selected user preference pairs are used for all values of λ.

This result provides further evidence that steering strength affects both task performance and personalization. It supports adjusting the strength at each generation step rather than using a single fixed value.

C Implementation Details

We describe the CCS implementation used in Section 3. The backbone is frozen, and one adapter is shared across users.

C.1 Prompt Templates and Preference History

We use the Qwen3 chat template with thinking disabled and no extra system message. Figure 6 shows the query-only and history-conditioned inputs. Each history example consists of a previous request paired with its preferred and rejected responses. We concatenate these examples with the current query into a single user message, and decoding starts after the empty `<think>` block in the assistant prefix.

Base receives only the current query, whereas ICL, CoS, SPL+RGD and CCS additionally receive the preference history. For CCS training, the query-only prompt supplies the context-free predictions used to construct the training targets. The preferred response to the current query serves only as the target output and is not included in the history.

(a) Query-only

```
<|im_start|>user
{x}<|im_end|>
<|im_start|>assistant
<think>

</think>
```

(b) History-conditioned

```
<|im_start|>user
Example 1
User request: {x_1}
Preferred assistant response: {y_1^+}
Less preferred assistant response: {y_1^-}

...

Example K
User request: {x_K}
Preferred assistant response: {y_K^+}
Less preferred assistant response: {y_K^-}

Current interaction:
{x}<|im_end|>
<|im_start|>assistant
<think>

</think>
```

Figure 6: **Query-only and context-conditioned prompts.** Braces mark text inserted for each example, and the ellipsis stands for additional history pairs. Both inputs use one user message and the same assistant prefix.

C.2 Architecture of CCS Adapter

Inputs and user context attention. The frozen LM separately encodes the ICL prompt $(\mathcal{H}_u, x)$ and the user context $\mathcal{H}_u$. Pooling and projection turn the ICL prompt states into a fixed prompt vector. For the user context memory, we split the user context states into at most 64 consecutive groups and average the states within each group. At each decoding step, an eight-head attention layer reads this memory using the current hidden state h_t and the fixed prompt vector. A small size encoder forms $\boldsymbol{z}_t$ from the projected current state and the user context feature returned by attention

Scores and effective gate. For each candidate token v, the score head predicts $s_t^\theta(v)$ using $\boldsymbol{z}_t$, the token's vector in the frozen LM output head, and features of the context-conditioned predictions. We subtract a weighted mean from the scores using

$$w_t(v) = \frac{p_t^{\mathrm{ctx}}(v)}{\sum_{u\in\mathbb{P}_t} p_t^{\mathrm{ctx}}(u)}, \qquad \mathcal{N}_t(r)(v) = r(v) - \sum_{u\in\mathbb{P}_t} w_t(u) r(u). \tag{11}$$

Thus, $\bar{s}_t^\theta = \mathcal{N}_t(s_t^\theta)$. We apply the same operation to the context direction d_t to obtain $\bar{d}_t$, without rescaling its variance.

The strength head predicts $\hat{\lambda}_t = 2\tanh(f_\lambda(\boldsymbol{z}_t))$, and the gate head predicts $g_t \in [0,1]$ with a sigmoid. The gate head also uses summary features of the candidate distribution. We combine these outputs into the effective gate shown in Fig. 1:

$$\bar{g}_t = -1 + g_t(\hat{\lambda}_t + 1), \qquad b_t(v) = \bar{g}_t\, \bar{s}_t^\theta(v). \tag{12}$$

We train g_t using the soft target $\tilde{g}_t$ and use $\bar{g}_t$ as the signed coefficient for steering. As g_t approaches zero, the coefficient approaches -1, and the adapter subtracts the learned steering direction from the context-conditioned logits. This aims to recover the context-free distribution, and does not guarantee an exact match.

Final correction. We control the size of the correction through smooth clipping, scaling, and a final weighted mean subtraction:

$$\delta_t^\theta(v) = \mathcal{N}_t(\alpha c \tanh(b_t/c))\,(v), \qquad v \in \mathbb{P}_t. \tag{13}$$

We apply this correction only to tokens in $\mathbb{P}_t$ and set it to zero if no user context is provided. The clipping parameter is fixed at $c = 2.0$. The scale α increases linearly from zero to the value in Table 5 over the first 10% of training updates and stays fixed for the remaining updates.

C.3 Plausible Tokens and Training Targets

Candidate selection. We sort tokens by their context-conditioned probability and select a set based on the distribution's uncertainty:

$$H_t = -\frac{\sum_{v\in\mathcal{V}} p_t^{\mathrm{ctx}}(v)\log p_t^{\mathrm{ctx}}(v)}{\log|\mathcal{V}|}, \qquad \rho_t = 0.70 + 0.25H_t. \tag{14}$$

Starting from the most likely token, we add candidates until their cumulative probability reaches ρ_t, subject to a minimum of 32 and a maximum of 128 tokens. For training, we extend this set with the preferred token and the top 32 tokens from the context-free distribution. These extra candidates are not added during generation.

Training targets and settings. To identify positions eligible for helpful distillation, we check that the preferred token belongs to $\mathbb{P}_t$ and that the weighted variance of the context direction is greater than 10^{-5}. The oracle strength is found by bisection over $[-2, 2]$ for 32 steps. For each KL loss, we normalize the target and student probabilities over the same candidate set. The target predictions and oracle strengths do not receive gradients and remain fixed during the adapter update. The main settings are listed in Table 5.

Local property of the oracle target. Fix a helpful position $t \in T_{\mathrm{help}}$, its preferred-response prefix, and its plausible token set P_t. Let q_t^{ctx} and q_t^{base} denote the context-conditioned and context-free distributions normalized over P_t, respectively. Mean-centering the context direction subtracts a token-independent constant, which cancels in the softmax. Therefore,

$$q_{t,\lambda=0} = q_t^{\mathrm{ctx}}, \qquad q_{t,\lambda=-1} = q_t^{\mathrm{base}}.$$

Since the oracle search interval $[-2, 2]$ contains both 0 and -1, any exact maximizer $\lambda_t^\star$ of $\log q_{t,\lambda}(y_t^+)$ over this interval satisfies

$$q_t^\star(y_t^+) \geq \max\left\{q_t^{\mathrm{ctx}}(y_t^+), q_t^{\mathrm{base}}(y_t^+)\right\}.$$

Thus, on the same candidate set, the exact oracle target assigns at least as much probability to the observed preferred token as either reference distribution. This is a local property of the training target, not a guarantee about the learned adapter or its generated responses. The finite-step oracle search numerically approximates this exact maximizer.

C.4 Algorithms

Algorithms 1 and 2 summarize training and generation.

Algorithm 1 Training Cautious Context Steering (CCS)

Require: Frozen LM f, adapter θ, data $\mathcal{D} = \{(x, y^+, \mathcal{H}_u)\}$

while training updates remain **do**
  Sample a minibatch and encode its ICL prompts and user contexts.
  Teacher-force y^+ with and without $\mathcal{H}_u$ to obtain ICL states h_t, p_t^{ctx}, and p_t^{base}.
  for each valid response position t **do**
    Build plausible token set $\mathbb{P}_t$ and compute advantage A_t and the context direction $\bar{d}_t$.
    Predict the distribution q_t^θ and gate g_t.
    if $t \in \mathcal{T}_{\mathrm{help}}$ **then**
      Find $\lambda_t^\star$ using Eq. 5 and form $q_t^\star$.
      $q_t^{\mathrm{target}} \leftarrow q_t^\star, \quad \tilde{g}_t \leftarrow \sigma((A_t - m)/\tau)$.
    else
      $q_t^{\mathrm{target}} \leftarrow q_t^{\mathrm{base}}, \quad \tilde{g}_t \leftarrow 0$.
    end if
  end for
  Compute $\mathcal{L}_{\mathrm{CCS}}$ using Eq. 10 and update θ.
end while

Algorithm 2 Generation with Cautious Context Steering (CCS)

Require: Frozen LM f, trained adapter θ, prompt x, user context $\mathcal{H}_u$

Encode the ICL prompt and user context.
while no EOS and below the length limit **do**
  $(h_t, \ell_t^{\mathrm{ctx}}) \leftarrow f(\mathcal{H}_u, x, y_{<t})$
  Select plausible set $\mathbb{P}_t$ from the context-conditioned predictions.
  Predict δ_t^θ from h_t and the stored features; set it to zero outside $\mathbb{P}_t$.
  Decode y_t from $\mathrm{softmax}(\ell_t^{\mathrm{ctx}} + \delta_t^\theta)$ and append it to y.
end while

Ensure: Personalized response y

Table 5: Main hyperparameters for CCS training and evaluation.

Hyperparameter	**Value**
Backbone models	Qwen3-0.6B, Qwen3-4B
Adapter and candidate dimensions	512
Strength and gate head hidden dimensions	256
Adapter dropout	0.05
Training epochs	4
Effective batch size	64
Optimizer	AdamW
Learning rate	2×10^{-4}
Learning rate schedule	Cosine decay with 3% warm-up
Weight decay	0.01
User context size	4 preference pairs
Oracle steering range	$[-2, 2]$
Helpfulness margin m	0.15
Gate target temperature τ	0.1
Loss weights (help, preserve, gate)	$(1.0, 0.7, 0.3)$
Score scale	1.0
Decoding strategy	Greedy
Maximum generated tokens	256
Training and Evaluation GPU	NVIDIA RTX A6000 $\times 2$, NVIDIA RTX 4090 $\times 4$

D Ablation Studies

We ablate each loss in Eq. 10 using Qwen3-4B on PRISM. For each variant, we set one loss weight to zero and keep all other settings unchanged. All runs use the same training data, initialization seed, four-epoch schedule, and evaluation settings. When $\mathcal{L}_{\text{gate}}$ is removed, the gate remains in the adapter and still receives gradients from the other losses. The full model achieves the highest scores on all three metrics in Table 6. Removing $\mathcal{L}_{\text{help}}$ lowers ROUGE-1 and ROUGE-L by 0.90 points each, the largest drops among the variants. Removing $\mathcal{L}_{\text{preserve}}$ leads to smaller drops in ROUGE, but lowers BERTScore-F1 by 0.56 points, close to the 0.54-point drop without $\mathcal{L}_{\text{help}}$. These results support using both oracle-steered targets at helpful positions and context-free targets at the remaining positions. Removing gate supervision also lowers all three scores, though the differences are small. Explicit gate supervision thus provides an additional gain even when the gate can learn from the other losses.

Table 6: Loss ablations on PRISM with Qwen3-4B.

Variant	R-1	R-L	B-F1
Full CCS	**36.73**	**21.51**	**87.12**
w/o $\mathcal{L}_{\text{help}}$	35.83	20.61	86.58
w/o $\mathcal{L}_{\text{preserve}}$	36.51	21.41	86.56
w/o $\mathcal{L}_{\text{gate}}$	36.54	21.37	87.05