DSPA: DYNAMIC SAE STEERING FOR DATA-EFFICIENT PREFERENCE ALIGNMENT

James Wedgwood, Aashiq Muhamed, Mona T. Diab, & Virginia Smith
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, PA 15213, USA
{jwedgwoo, amuhamed, mdiab, smithv}@cmu.edu

ABSTRACT

Preference alignment is usually achieved by weight-updating training on preference data, which adds substantial alignment-stage compute and provides limited mechanistic visibility. We propose Dynamic SAE Steering for Preference Alignment (DSPA), an inference-time method that makes sparse autoencoder (SAE) steering prompt-conditional. From preference triples, DSPA computes a conditional-difference map linking prompt features to generation-control features; during decoding, it modifies only token-active latents, without base-model weight updates. Across Gemma-2-2B/9B and Qwen3-8B, DSPA improves MT-Bench and is competitive on AlpacaEval while preserving multiple-choice accuracy. Under restricted preference data, DSPA remains robust and can rival the two-stage RAHF-SCIT pipeline while requiring up to $4.47\times$ fewer alignment-stage FLOPs. Finally, we audit the SAE features DSPA modifies, finding that preference directions are dominated by discourse and stylistic signals, and provide theory clarifying the conditional-difference map estimate and when top-k ablation is principled. [1]

1 INTRODUCTION

Preference alignment is central to deploying large language models (LLMs), but common pipelines rely on weight-updating training on preference data (e.g., RLHF, DPO (Rafailov et al., 2023)), which can be costly and mechanistically opaque. Inference-time activation interventions (Zou et al., 2023; Kong et al., 2024; Pham & Nguyen, 2024) are cheaper and reversible, but are often defined as dense, global directions; applying the same edit in every context can degrade open-ended generation and yield hard-to-audit tradeoffs (Wolf et al., 2024).

Sparse autoencoders (SAEs) offer a more inspectable coordinate system: they decompose hidden states into sparse, named features that can be directly ablated or amplified (Rimsky et al., 2024; Durmus et al., 2024). However, preference alignment for open-ended generation is prompt-dependent. Static SAE feature sets can inject context-irrelevant signal and harm dialogue quality (Bayat et al., 2025a), motivating prompt-conditional feature selection and minimal, targeted edits.

We propose **Dynamic SAE Steering for Preference Alignment (DSPA)**, a prompt-conditional inference-time method that requires no base-model weight updates. DSPA uses an early–mid-layer *input SAE* to featurize the prompt and a late-layer *output SAE* as the intervention space most likely to influence generation (Arad et al., 2025). Offline, we compute a sparse conditional-difference map $\mathbf{A}$ from preference triples that associates active prompt features with output features that differ between chosen and rejected responses. At inference, DSPA selects a small set of prompt features, scores output features via $\mathbf{A}$, and steers decoding by ablating and/or augmenting only those output latents that are active on the current token, reducing off-context perturbations relative to static steering.

We evaluate DSPA on Gemma-2-2B/9B and Qwen3-8B (Team et al., 2024; Yang et al., 2025) using two open-ended generation benchmarks, MT-Bench (Zheng et al., 2023) and AlpacaEval (Li et al., 2023), and multiple-choice benchmarks. We treat open-ended judge scores as our primary alignment

[1] Code for this paper is available at https://github.com/jtbwedgwood/dspa.

utility metric, track multiple-choice accuracy for capability preservation, and quantify alignment-stage compute/memory and preference-data requirements (N) as alignment cost. DSPA improves MT-Bench across all three models; on AlpacaEval, it improves on Gemma-2-2B and Qwen3-8B, with modest changes on multiple-choice metrics. DSPA remains robust under restricted preference data (e.g., 250 samples; stable down to 100 on Gemma-2-2B) and can rival the two-stage RAHF-SCIT pipeline while requiring up to $4.47\times$ fewer alignment-stage FLOPs. Because we intervene on named SAE features, we can audit what changes and find that preference directions are dominated by discourse and stylistic signals; our theory characterizes what $\mathbf{A}$ estimates and when top-k ablation is a principled selection rule. Our contributions include:

1. We introduce DSPA, a prompt-conditional inference-time method that constructs a sparse association map from prompt SAE features (early/mid-layer) to generation-control SAE features (late-layer) via a conditional-difference map, and steers decoding by editing only token-active latents.
2. We evaluate DSPA across Gemma-2-2B/9B and Qwen3-8B on open-ended generation benchmarks and multiple-choice benchmarks, showing competitive generation quality with minimal capability regression and robustness under preference-data restriction. DSPA can rival the two-stage RAHF-SCIT pipeline while requiring up to $4.47\times$ fewer alignment-stage FLOPs.
3. We audit the SAE features DSPA modifies and find that preference gains are largely mediated by discourse and stylistic signals; we also provide a theoretical analysis of what the conditional-difference map estimates and why top-k ablation is a principled selection rule.

2 Background and Related Work

Sparse Autoencoders. Superposition suggests many features share directions in activation space (Elhage et al., 2022). Sparse autoencoders (SAEs) learn a sparse feature vector and a linear reconstruction of a layer's activations (Cunningham et al., 2023; Gao et al., 2024a). Concretely, given activations $\mathbf{h} \in \mathbb{R}^{d_{\text{model}}}$, an SAE encodes $\mathbf{f}(\mathbf{h}) = \sigma(W_{\text{enc}}\mathbf{h} + \mathbf{b}_{\text{enc}})$ and decodes $\hat{\mathbf{h}}(\mathbf{f}) = W_{\text{dec}}\mathbf{f} + \mathbf{b}_{\text{dec}}$, trained to minimize reconstruction error with a sparsity penalty. We refer to the activation of feature i as $f_i(\mathbf{h})$; sparse features enable auditable interventions by editing a small set of latents and decoding back to hidden-state space (Durmus et al., 2024).

Preference Alignment. Most preference alignment methods update model weights using preference data (e.g., RLHF, DPO (Rafailov et al., 2023)), which can be effective but expensive and opaque. Inference-time interventions instead steer activations and are reversible. Representation engineering extracts linear control directions (mean-difference/PCA-style) and exposes alignment–capability tradeoffs (Zou et al., 2023; Wolf et al., 2024), while RAHF learns activity patterns from chosen/rejected data (Liu et al., 2024). Recent work refines steering with geometric and control-theoretic views (Kong et al., 2024; Pham & Nguyen, 2024) and studies category-specific directions for safety (Bhattacharjee et al., 2024). DSPA builds on this inference-time line but makes steering prompt-conditional and operates in a sparse SAE basis for auditability.

SAEs for Preference Alignment. SAE steering has been used for refusal and safety control (Rimsky et al., 2024), and static feature sets can work in constrained settings such as multiple-choice calibration (Bayat et al., 2025a). Feature selection remains central: effective steering depends on causal influence, not just activation frequency (Arad et al., 2025). For open-ended generation, prior approaches often rely on learned steering policies or direct optimization in SAE space (Ferrao et al., 2025a; Bounhar et al., 2026). DSPA instead computes a map from prompt features to response-control features without training a steering policy or updating base-model weights.

3 Dynamic SAE Steering for Preference Alignment

DSPA is a two-stage inference-time method. In an offline stage, we compute a sparse *conditional-difference map* $\mathbf{A}$ from preference triples, associating prompt features with response-control features that differentiate chosen from rejected continuations. At inference, we use the prompt's active features

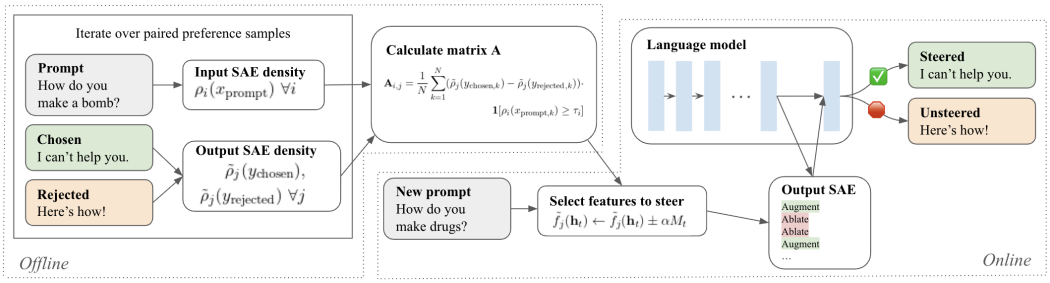

Figure 1: DSPA overview: offline conditional-difference map construction and prompt-conditional SAE steering at inference. Prompts and responses are illustrative. DSPA replaces costly weight updates with prompt- and token-conditional, directly auditable SAE feature edits.

to retrieve and rank a small set of output features and steer decoding by ablating and/or augmenting only token-active latents in the output SAE. Because interventions occur on named SAE features, every edit is directly auditable.

Relation to representation engineering. DSPA shares the use of chosen/rejected pairs to estimate preference directions (Zou et al., 2023; Liu et al., 2024). If we ignore prompt conditioning and steer along a single dense direction, the construction reduces to global mean-difference steering. DSPA instead builds a prompt-conditional library of sparse templates (via input-feature gating) and applies *token-conditional* edits in an SAE basis, modifying only latents that are active on the current token.

3.1 Identifying Conditional Preference Features

Why two SAEs? The features that are useful to *condition on* need not be the best ones to *steer*. Following prior work suggesting that late-layer SAE features have stronger causal influence on generation (Arad et al., 2025), we use an early–mid-layer *input SAE* to featurize the prompt and a late-layer *output SAE* as the intervention space; we validate this choice empirically (Section 4.5).

Notation. Let $\mathcal{D} = \{(x_k, y_k^+, y_k^-)\}_{k=1}^N$ be a dataset of preference triples, where x is a prompt and y^+, y^- are the chosen and rejected responses. Let d_{SAE} be the SAE width (we assume the same width for input and output SAEs). We index input-SAE features by i and output-SAE features by j, and we use a tilde to distinguish output-SAE quantities. Let $\mathbf{h}_t^{\text{in}}$ denote the LLM hidden state at token position t at the input-SAE layer, and $\mathbf{h}_t^{\text{out}}$ the hidden state at the output-SAE layer. Then $f_i(\mathbf{h}_t^{\text{in}})$ denotes input-SAE feature i and $\tilde{f}_j(\mathbf{h}_t^{\text{out}})$ denotes output-SAE feature j.

Activation densities. For a prompt $x = (x_1, \ldots, x_{T_x})$ and continuation $y = (y_1, \ldots, y_{T_y})$ (evaluated under teacher forcing after x), define

$$\rho_i(x) := \frac{1}{T_x}\sum_{t=1}^{T_x} \mathbf{1}\left[f_i(\mathbf{h}_t^{\text{in}}) > 0\right],$$

$$\tilde{\rho}_j(x, y) := \frac{1}{T_y}\sum_{t=1}^{T_y} \mathbf{1}\left[\tilde{f}_j(\mathbf{h}_{T_x+t}^{\text{out}}) > 0\right].$$

That is, $\rho_i(x)$ is the fraction of prompt tokens activating input feature i, and $\tilde{\rho}_j(x, y)$ is the fraction of response tokens activating output feature j when processing the continuation y after x.

Prompt gates. Choose a percentile $p \in (0, 100)$ and let τ_i be the pth percentile of $\rho_i(x)$ over prompts in $\mathcal{D}$. Define the gate $g_i(x) := \mathbf{1}[\rho_i(x) \geq \tau_i]$ and the gate vector $g(x) \in \{0, 1\}^{d_{\text{SAE}}}$.

Conditional-difference map. Let us define $\Delta\tilde{\boldsymbol{\rho}}(x, y^+, y^-) := \tilde{\boldsymbol{\rho}}(x, y^+) - \tilde{\boldsymbol{\rho}}(x, y^-)$, where $\tilde{\boldsymbol{\rho}}(x, y) \in [0, 1]^{d_{\text{SAE}}}$ stacks $\tilde{\rho}_j(x, y)$. Write $\Delta\tilde{\boldsymbol{\rho}}_k := \Delta\tilde{\boldsymbol{\rho}}(x_k, y_k^+, y_k^-)$ and compute $\mathbf{A} := \frac{1}{N}\sum_{k=1}^N g(x_k)\,\Delta\tilde{\boldsymbol{\rho}}_k^\top$, where $\mathbf{A} \in \mathbb{R}^{d_{\text{SAE}} \times d_{\text{SAE}}}$.

Entrywise, $\mathbf{A}_{i,j}$ averages the chosen-minus-rejected activation-density difference for output feature j over examples where prompt gate i is active, so $\mathbf{A}_{i,j}$ estimates $\mathbb{E}[g_i(x)\,\Delta\tilde{\rho}_j]$. Note that $\mathbf{A}$ is not a learned reward model or steering policy; it is a fixed association map requiring no gradient-based optimization.

Sparsification. Although $\mathbf{A}$ has d_{SAE}^2 entries, it is sparse in practice. We zero out entries below a conservative threshold, reducing memory usage by 99.8% (Appendix B).

3.2 Inference-Time Intervention

Prompt feature selection. Given a new prompt x, we compute $\rho_i(x)$ for all input features and select the top k_{prompt} features by density, $S_{\text{prompt}}(x) := \text{top-}k_{\text{prompt}}\{\rho_i(x)\}$. Let $\hat{g}(x) := \mathbf{1}_{S_{\text{prompt}}(x)} \in \{0,1\}^{d_{\text{SAE}}}$, where $\mathbf{1}_S$ denotes the indicator vector of a set S.

Scoring output features. We score output features using the conditional-difference map, $\mathbf{s}(x) := \mathbf{A}^\top \hat{g}(x)$. We then select k_{diff} features to augment and/or ablate: $S_{\text{augment}}(x) := \text{top-}k_{\text{diff}}\{\mathbf{s}_j(x)\}$ and $S_{\text{ablate}}(x) := \text{bottom-}k_{\text{diff}}\{\mathbf{s}_j(x)\}$.

Token-conditional intervention. $S_{\text{augment}}(x)$ and $S_{\text{ablate}}(x)$ specify *which* output features to steer for prompt x; token-conditional steering edits only features active on the current token, avoiding off-context activations.

At token position t, we compute output-SAE latents $\tilde{\mathbf{f}}_t = \tilde{\mathbf{f}}(\mathbf{h}_t^{\text{out}})$ and scale the step size by $M_t := \max_{j\in[d_{\text{SAE}}]} \tilde{f}_{t,j}$. For $j \in S_{\text{augment}}(x)$ with $\tilde{f}_{t,j} > 0$, we set $\tilde{f}_{t,j} \leftarrow \tilde{f}_{t,j} + \alpha M_t$; for $j \in S_{\text{ablate}}(x)$ with $\tilde{f}_{t,j} > 0$, we set $\tilde{f}_{t,j} \leftarrow \max\{\tilde{f}_{t,j} - \alpha M_t, 0\}$, yielding edited latents $\tilde{\mathbf{f}}'_t$. Let $\Delta\tilde{\mathbf{f}}_t := \tilde{\mathbf{f}}'_t - \tilde{\mathbf{f}}_t$; we apply the residual edit by $\mathbf{h}_t^{\text{out}} \leftarrow \mathbf{h}_t^{\text{out}} + \tilde{W}_{\text{dec}}\,\Delta\tilde{\mathbf{f}}_t$. Because our SAEs use ReLU activations, this edits only already-active latents and clamps ablations at zero. In our main experiments we use ablation-only steering unless otherwise stated.

3.3 Theoretical Analysis

The preceding procedure is heuristic; we now provide formal justification for why the conditional-difference map recovers meaningful preference directions and why top-k selection is principled. Proofs appear in Appendix A.

Setup. We use the notation from Section 3.1: x is the prompt and y^+, y^- are the chosen/rejected responses. For a prompt x, let $g(x) \in \{0,1\}^{d_{\text{SAE}}}$ denote the *prompt gate vector* with $g_i(x) := \mathbf{1}[\rho_i(x) \geq \tau_i]$, where $\rho_i(x)$ is the input-SAE activation density and τ_i is a fixed threshold. For a response y generated after x, let $\tilde{\boldsymbol{\rho}}(y) \in [0,1]^{d_{\text{SAE}}}$ denote output-SAE activation densities over the response segment (equivalently $\tilde{\boldsymbol{\rho}}(x,y)$ in Section 3.1, suppressing x for brevity). Given a preference triple (x, y^+, y^-), define $\Delta\tilde{\boldsymbol{\rho}} := \tilde{\boldsymbol{\rho}}(y^+) - \tilde{\boldsymbol{\rho}}(y^-)$. The population analogue of our conditional-difference map is $\mathbf{A} := \mathbb{E}[g(x)\,\Delta\tilde{\boldsymbol{\rho}}^\top]$.

Assumption 3.1 (Shared-covariance mean shift)**.** There exist a positive semidefinite matrix Σ, a scalar $c > 0$, and a prompt-dependent vector $\boldsymbol{\beta}(x)$ such that, conditional on x, $\mathbb{E}[\Delta\tilde{\boldsymbol{\rho}} \mid x] = c\,\Sigma\boldsymbol{\beta}(x)$.

Assumption 3.2 (Additive gating)**.** There exists a matrix $B \in \mathbb{R}^{d_{\text{SAE}}\times d_{\text{SAE}}}$ such that $\boldsymbol{\beta}(x) = B\,g(x)$. Let $M := \mathbb{E}[g(x)\,g(x)^\top]$ denote the gate Gram matrix.

Theorem 3.3 (Factorization of $\mathbf{A}$)**.** *Under Theorems 3.1 and 3.2, $\mathbf{A}^\top = c\,\Sigma\,B\,M$. Consequently, rows of $\mathbf{A}$ mix multiple preference templates when gates co-activate. On subspaces where M is invertible, $\Sigma^{-1}\mathbf{A}^\top M^{-1} \propto B$.*

The mixing is controlled when co-activation probabilities are small:

Corollary 3.4 (Weak co-activation bound)**.** *Fix gate i and let $\pi_{i'|i} := \Pr(g_{i'} = 1 \mid g_i = 1)$. Then $\|\mathbb{E}[\Delta\tilde{\boldsymbol{\rho}} \mid g_i = 1] - c\,\Sigma\boldsymbol{\beta}^{(i)}\|_2 \leq c\,\|\Sigma\|_{2\to 2} \sum_{i'\neq i} \pi_{i'|i} \|\boldsymbol{\beta}^{(i')}\|_2$.*

Connection to DSPA. At inference, DSPA computes $\mathbf{s}(x) := \mathbf{A}^\top \hat{g}(x)$ where $\hat{g}(x)$ is a truncated indicator over active prompt features. By Theorem 3.3, $\mathbf{s}(x) = c\,\Sigma\,B\,M\,\hat{g}(x)$, a mixed estimate of

the prompt-conditional preference direction. When $M \approx \mathrm{diag}(p)$, this reduces to a reweighted sum of per-gate templates. Under a linear utility model, selecting output features with the most negative scores for ablation is optimal for a fixed budget (Theorem A.4; Appendix A).

Implications. These results justify several design choices. First, the factorization $\mathbf{A}^\top = c\,\Sigma\,B\,M$ shows that $\mathbf{A}$ is a principled estimator of prompt-conditional preference directions even though it is constructed by simple averaging, not optimization. Second, Theorem 3.4 explains why truncation to the top k_{prompt} gates is reasonable: when prompt features activate approximately independently, each row of $\mathbf{A}$ approximates a clean per-gate preference template with bounded contamination from co-activating gates. This also clarifies why static (non-prompt-conditional) steering is brittle for open-ended generation: it ignores co-activation structure entirely. Third, Theorem A.4 justifies top-k ablation as the optimal selection rule under a linearized utility model. Finally, the finite-sample concentration result (Theorem A.5, Appendix A) connects to our data-efficiency findings: frequently activated gates yield stable conditional estimates even with limited data, while rare gates are noisy, motivating our percentile thresholding and conservative sparsification of $\mathbf{A}$.

4 Experiments and Results

4.1 Models, Data, and SAEs

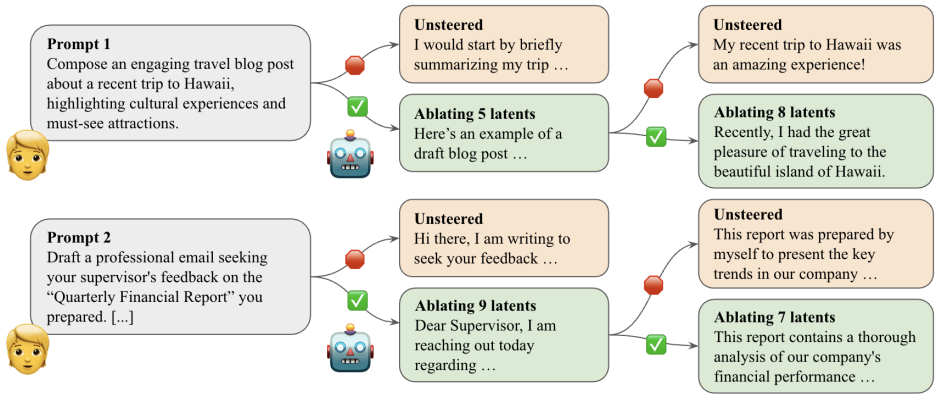

Figure 2: MT-Bench (turn 1) example on Gemma-2-9B: Base Model vs. DSPA. DSPA can improve open-ended response quality over the SFT base without weight updates.

Figure 2 shows a qualitative example illustrating how DSPA changes open-ended responses.

Training Data We use a binarized version of UltraFeedback (Cui et al., 2024) (61K prompts with chosen/rejected responses).[2] We use these preference pairs to construct $\mathbf{A}$ and, where applicable, to train weight-updating baselines (DPO/RAHF) and derive inference-time steering controls (RepE direction; Static-SAE feature set).

Models Following common alignment setups (Ouyang et al., 2022; Liu et al., 2024), we start from supervised fine-tuned (SFT) base models on HH-RLHF (Bai et al., 2022) for Gemma-2-2B/9B and Qwen3-8B (Team et al., 2024; Yang et al., 2025).[3] We use these architectures because high-quality open SAEs are available; "Base Model" refers to the HH-RLHF SFT checkpoint.

SAEs We use SAEs fine-tuned on HH-RLHF (Appendix G), which yields better-aligned features. For Gemma, we fine-tune Gemma Scope JumpReLU SAEs (Lieberum et al., 2024; Rajamanoharan et al., 2024); for Qwen, we use released BatchTopK SAEs (Bussmann et al., 2024).[4]

[2] HF: `argilla/ultrafeedback-binarized-preferences-cleaned`.
[3] HF: `Anthropic/hh-rlhf`.
[4] HF: `adamkarvonen/qwen3-8b-saes`.

Table 1: Open-ended and multiple-choice results (higher is better). Open-ended scores are averaged over three generation seeds; multiple-choice scores are deterministic. Bold = best; underlined = second best within each model group. Arc = Arc-Easy; TQA = TruthfulQA-MC2. DSPA improves MT-Bench across all three models with minimal multiple-choice regression, outperforming other inference-time baselines. See Appendix D.

		Open-ended		*Multiple-choice*				
		MT-Bench	**AlpacaEval**	**MMLU**	**Arc**	**TQA**	**HellaSwag**	**Winogrande**
Gemma-2-2B	DSPA (ours)	**4.39**	**19.30**	**48.1**	80.4	40.4	74.2	65.0
	DPO	<u>4.36</u>	15.82	47.9	**81.0**	39.8	**74.3**	65.2
	RepE	4.32	16.94	48.0	80.3	40.3	**74.3**	65.1
	Static-SAE	3.49	9.33	48.0	80.6	39.5	74.2	**65.7**
	Prompt Eng	3.84	15.28	47.8	80.8	**41.0**	74.0	64.3
	Base Model	3.91	<u>17.54</u>	47.9	80.4	40.2	74.2	**65.7**
Gemma-2-9B	DSPA (ours)	<u>5.02</u>	20.00	**60.6**	**83.8**	45.4	77.3	**72.5**
	DPO	**5.05**	17.39	60.4	83.6	44.1	77.3	**72.5**
	RepE	4.36	18.88	60.4	**83.8**	45.5	77.5	**72.5**
	Static-SAE	4.33	13.56	**60.6**	83.6	44.1	**77.6**	72.5
	Prompt Eng	4.60	**22.51**	60.0	83.7	**47.2**	77.4	71.9
	Base Model	4.39	<u>21.12</u>	60.0	83.7	45.5	77.4	71.9
Qwen3-8B	DSPA (ours)	<u>4.86</u>	<u>17.83</u>	71.2	84.3	42.4	72.6	73.8
	DPO	**5.52**	**24.60**	**72.6**	**85.5**	**45.9**	**73.0**	73.4
	RepE	4.48	12.95	71.2	84.4	41.0	71.7	73.1
	Static-SAE	3.24	8.58	71.2	84.1	42.4	72.7	**74.0**
	Prompt Eng	3.88	11.74	71.0	84.4	42.6	71.7	72.8
	Base Model	4.12	13.47	71.1	84.4	40.9	71.6	72.0

Baselines We include the HH-RLHF supervised fine-tuned starting point ("Base Model") as a reference, and compare DSPA to four baselines: DPO (Rafailov et al., 2023) fine-tuned on UltraFeedback chosen/rejected pairs (Cui et al., 2024), a dense representation-engineering control direction (RepE) estimated from chosen/rejected data (Zou et al., 2025), a static SAE steering baseline that edits a fixed globally selected feature set (Static-SAE) (Ferrao et al., 2025b; Bayat et al., 2025b), and an instruction-style prompt prefix (Prompt Eng) (Liu et al., 2024; Wu et al., 2025). DPO updates base-model weights, whereas DSPA/RepE/Static-SAE are inference-time activation interventions. In Section 4.4, we additionally compare against RAHF-SCIT (Liu et al., 2024), a two-stage fine-tuning pipeline, under a restricted-data setting. Full hyperparameters and implementation details are in Appendix D.2.

4.2 Main Results: Open-Ended and Multiple-Choice Benchmarks

We evaluate open-ended generation with MT-Bench (Zheng et al., 2023) and AlpacaEval (Li et al., 2023), both using LLM-as-a-judge protocols. We use GPT-4o as the MT-Bench judge; for AlpacaEval, we use the `alpaca_eval_llama3_70b_fn` annotator, so absolute AlpacaEval scores are not directly comparable to leaderboards that use different judges. To monitor general capability, we report multiple-choice benchmarks (MMLU, Arc-Easy, TruthfulQA-MC2, HellaSwag, Winogrande) via the Language Model Evaluation Harness (Gao et al., 2024b) using standard settings (Appendix D).

Table 1 summarizes results (open-ended scores averaged over three generation seeds; small differences should be interpreted cautiously). On MT-Bench, DSPA improves over the Base Model and over other inference-time baselines for all three models, and it matches or slightly exceeds DPO on Gemma-2-2B without weight updates. On AlpacaEval, DSPA improves over the Base Model on Gemma-2-2B and Qwen3-8B, but underperforms both Prompt Eng and the Base Model on Gemma-2-9B. Static-SAE consistently degrades open-ended scores, supporting the need for prompt-conditional feature selection rather than a fixed global feature set. Multiple-choice metrics change only modestly across methods, suggesting limited capability regression. In Appendix E, we further break down MT-Bench scores by category.

Table 2: Top five ablate/augment-set features, with `gpt-5-mini` interpretations. Highly ranked preference features are dominated by discourse and stylistic cues.

	Feature	Interpretation
Ablate	11569	Filler phrases used to smooth conversation and request clarification
	10776	Questions or statements involving illegal or clearly harmful activities
	6345	Polite opening phrases that frame what follows
	11287	First-person statements expressing personal views or experiences
	12550	Short connective phrases linking ideas across clauses
Augment	141	Common function words and grammatical connectors without semantic content
	9825	Clarification and intent-seeking phrases linking questions and follow-ups
	10075	Topic-naming or subject-introducing words and phrases
	1030	Short list items and connectors enumerating entities or phrases
	11031	Affirmative explanatory openings that introduce a helpful continuation

Table 3: Category breakdown for ablate/augment sets (50 each) vs. 50 random SAE features. DSPA's frequently steered features concentrate on discourse/style rather than topical content.

Category	Ablate	Augment	Random
Content	5	7	18
Discourse	34	28	15
Grammatical	10	10	11
Structure	1	5	6

4.3 Anatomy of Preference in SAE Space

A distinctive advantage of DSPA over dense steering and fine-tuning methods is that every intervention is directly inspectable: because we operate on named SAE features, we can characterize the latent factors that drive preference gains. This analysis is consistent with the theoretical expectation from Section 3.3: because $\mathbf{A}$ averages over many prompts, the features that emerge most strongly are high-frequency, broadly reusable interactional signals rather than prompt-specific content.

We approximate frequently steered output features by ranking columns j of $\mathbf{A}$ by $\sum_i \mathbf{A}_{i,j}$: the top/bottom 50 form "augment"/"ablate" sets (Gemma-2-9B, $\ell_{\text{output}} = 39$). We find that these account for most features steered in practice: on the first turn of MT-Bench, the ablated features form a strict subset of our ablate set. In an auxiliary augment+ablate pass used only for this coverage check, all but four augmented features belong to the augment set.

We generate an explanation for each of these features via an interpretability pipeline using `gpt-5-mini` as a judge (see Appendix G.1). As these interpretations are LLM-generated and have not been systematically validated by humans, they should be treated as approximate. Descriptions of the top five augment and ablate set features, ordered by magnitude $|\sum_i \mathbf{A}_{i,j}|$, appear in Table 2. We also prompt `gpt-5-mini` to categorize each feature into one of four categories:

1. **Content:** Topic-bearing or referential language with substantive semantic meaning
2. **Discourse:** Language that manages conversational flow or interaction rather than content
3. **Grammatical:** Grammatical function words with minimal standalone meaning
4. **Structure:** Tokens whose primary role is positional or structural rather than semantic

Table 3 shows the breakdown by category for augment and ablate sets, compared to a sample of 50 random features as a baseline. Preference directions are dominated by discourse and style markers: both the augment and ablate sets contain far more Discourse and far fewer Content features than the random set. This is consistent with the view that preference-relevant signals for open-ended generation are primarily interactional — tone, hedging, clarification — rather than about specific concepts. Notably, relatively few features in either set are Structure features, indicating that DSPA relies on conversational signals rather than formatting like lists or code. This contrasts with another recent SAE steering method, FSRL (Ferrao et al., 2025b), which primarily relies on features related to "structural presentation elements." Safety-specific signal is limited: the only directly safety-related

Table 4: Restricted-data results (N=250 preference triples; higher is better). Open-ended scores are averaged over three generation seeds; multiple-choice scores are deterministic. Bold = best; underlined = second best within each model. 2B = Gemma-2-2B; 9B = Gemma-2-9B; Qwen = Qwen3-8B. Under severe data restriction, DSPA remains competitive with the two-stage RAHF-SCIT pipeline while using far less alignment compute (Appendix C).

		Open-ended		*Multiple-choice*				
		MT-Bench	**AlpacaEval**	**MMLU**	**Arc**	**TQA**	**HellaSwag**	**Winogrande**
2B	DSPA	3.95	16.81	**48.0**	80.4	40.4	74.2	65.0
	RAHF-SCIT	3.71	**19.13**	45.8	78.7	**40.6**	68.8	**66.3**
	DPO	3.91	15.42	47.9	**80.8**	39.6	74.0	64.6
	RepE	**3.97**	15.80	**48.0**	80.5	40.3	**74.3**	65.0
9B	DSPA	**4.82**	21.27	**60.9**	83.8	45.4	77.5	72.5
	RAHF-SCIT	4.63	**22.64**	58.8	82.6	45.4	73.6	72.4
	DPO	4.10	16.52	60.5	**84.6**	44.8	**77.6**	**72.8**
	RepE	4.47	20.62	60.5	83.9	**45.5**	77.3	72.4
Qwen	DSPA	**4.69**	18.20	71.1	84.3	42.3	**72.5**	**73.7**
	RAHF-SCIT	4.52	**23.48**	**71.3**	**84.6**	**48.2**	72.4	73.0
	DPO	4.34	16.84	71.3	84.5	41.1	71.7	72.4
	RepE	4.51	14.34	71.2	84.2	41.0	71.6	72.1

feature is the second-most ablated one, index 10776, representing *Questions or statements involving illegal or clearly harmful activities*.

4.4 Data Efficiency and Compute Cost

To study data efficiency, we subsample N=250 preference triples from UltraFeedback and recompute $\mathbf{A}$; for weight-updating methods, we train on the same subset. We compare against DPO and RepE, as well as RAHF-SCIT (Liu et al., 2024), a strong but computationally heavy two-stage fine-tuning pipeline. Using standard FLOP accounting (Brown et al., 2020; Chowdhery et al., 2022), we estimate that the alignment stage of RAHF-SCIT is approximately $4.47\times$ as compute-intensive as DSPA's $\mathbf{A}$ construction. In practice, on Gemma-2-9B on a single Nvidia H200, RAHF-SCIT takes $11.5\times$ longer wall-clock with peak memory 140.8 GB versus 33.1 GB (Appendix C). This comparison counts only additional alignment computation beyond the shared SFT base.

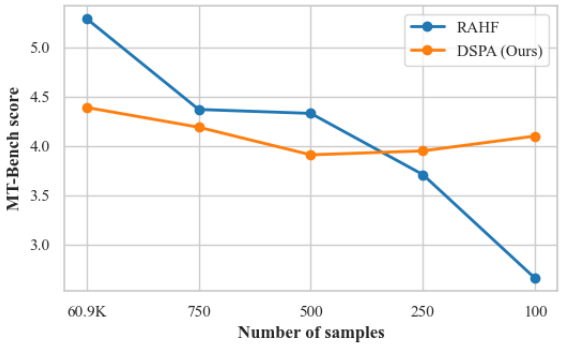

Figure 3: MT-Bench score (higher is better) vs. preference triples N under data restriction (Gemma-2-2B): DSPA vs. RAHF-SCIT. DSPA degrades gracefully down to N=100, while RAHF-SCIT drops sharply.

Table 4 reports results at N=250. In this setting, DSPA achieves the highest MT-Bench scores on Gemma-2-9B and Qwen3-8B and is within 0.02 of the best method on Gemma-2-2B. Both DSPA and RAHF-SCIT outperform DPO and RepE under data restriction. Figure 3 further sweeps N for Gemma-2-2B: DSPA remains stable even with as few as N=100 preference triples, while RAHF-SCIT's score declines sharply. This robustness is consistent with the concentration analysis in Appendix A: because the most frequently steered features (Section 4.3) are broadly reusable discourse signals, their conditional estimates in $\mathbf{A}$ stabilize quickly even with limited data.

4.5 Ablations

In the preceding experiments, we selected a consistent set of hyperparameters that yields strong results across all base models. In particular, we use $k_{\text{prompt}} = 32, k_{\text{diff}} = 16, \alpha = 0.2$, and we ablate dispreferred SAE features as opposed to augmenting preferred ones. We set $\ell_{\text{input}} = 7, \ell_{\text{output}} = 24$ for Gemma-2-2B, $\ell_{\text{input}} = 9, \ell_{\text{output}} = 39$ for Gemma-2-9B, and $\ell_{\text{input}} = 9, \ell_{\text{output}} = 18$ for Qwen3-8B. Moreover, as discussed in Section 4.1, we use custom SAEs fine-tuned on HH-RLHF. In this section, we justify our choice of input and output layer, mode (ablate vs. augment), and the use of custom SAEs in a series of ablation experiments run on the two Gemma models.

For ablations, we use MT-Bench judged by GPT-OSS-120B (much cheaper than GPT-4o); these scores are not directly comparable to our main MT-Bench results and are generally lower.

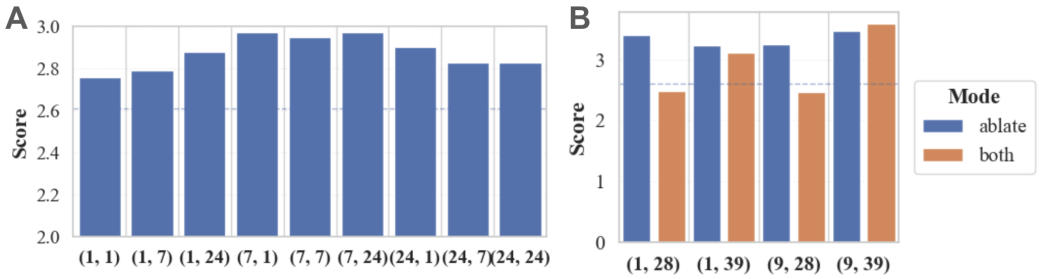

Figure 4: Layer choice ablations (MT-Bench with GPT-OSS-120B judge; higher is better). **A.** Gemma-2-2B score vs. $(\ell_{\text{input}}, \ell_{\text{output}})$ (ablate only). **B.** Gemma-2-9B score vs. $(\ell_{\text{input}}, \ell_{\text{output}})$ (ablate only; augment+ablate). Dashed = Base Model. An early–mid input layer and a late output layer yield the strongest scores.

Which layers matter? We examine the effect of different choices of ℓ_{input} and ℓ_{output} in Figure 4A for Gemma-2-2B and in Figure 4B for Gemma-2-9B. We observe that for Gemma-2-2B ℓ_{input} has a stronger impact on score than ℓ_{output}, with $\ell_{\text{input}} = 7$ outperforming the other input layers surveyed. $\ell_{\text{input}} = 7, \ell_{\text{output}} = 24$ has marginally better performance than other output layers choices; we also prefer a late output layer because late-layer SAE features are more likely to be interpretable.

For Gemma-2-9B, we observe stronger dependence on both input and output layer, particularly when both augmenting and ablating features. The best score is achieved in both cases for $\ell_{\text{input}} = 9, \ell_{\text{output}} = 39$. This is consistent with our choice of layers for Gemma-2-2B: in both cases we use an early–mid input layer and a late output layer.

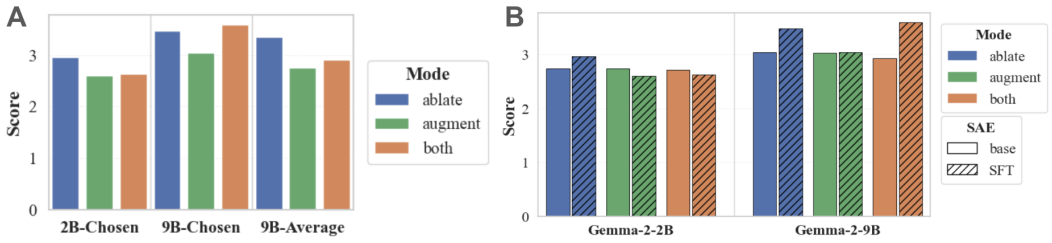

Figure 5: Steering-mode and SAE ablations (MT-Bench with GPT-OSS-120B judge; higher is better). **A.** Score by steering mode (ablate/augment/both). 2B-Chosen and 9B-Chosen use the best $(\ell_{\text{input}}, \ell_{\text{output}})$ per model; 9B-Average averages over the layer grid in Figure 4B. **B.** Base Gemma Scope SAE vs. HH-RLHF fine-tuned SAE. Ablation-only is most reliable, and finetuned SAEs improve steering performance.

Ablate, augment, or both? A key decision point in applying DSPA is whether to augment desirable features, ablate undesirable ones, or do both; Figure 5A summarizes the impact of this choice. We find that for both models only augmenting features leads to subpar results. For Gemma-2-2B, only

ablating features is consistently better than either of the other modes. For Gemma-2-9B, under certain layer combinations including our main choice of $\ell_{\text{input}} = 9, \ell_{\text{output}} = 39$, both augmenting and ablating yields a better score than ablating only, but this is not consistent across layer choices. For consistency, we only ablate features throughout.

Does fine-tuning the SAE help? In Figure 5B, we assess the impact of steering the base Gemma Scope SAEs (Lieberum et al., 2024) versus the custom SAEs fine-tuned on HH-RLHF which we have used throughout. We report results across both models and all three modes, keeping all other parameters constant. We observe smaller variance in score for the base SAE across modes, with performance comparable to the fine-tuned SAE for Gemma-2-2B and for Gemma-2-9B when augmenting features. Conversely, the advantage of the fine-tuned SAE is strongest when using Gemma-2-9B and either ablating or both augmenting and ablating features. See Appendix B for more details.

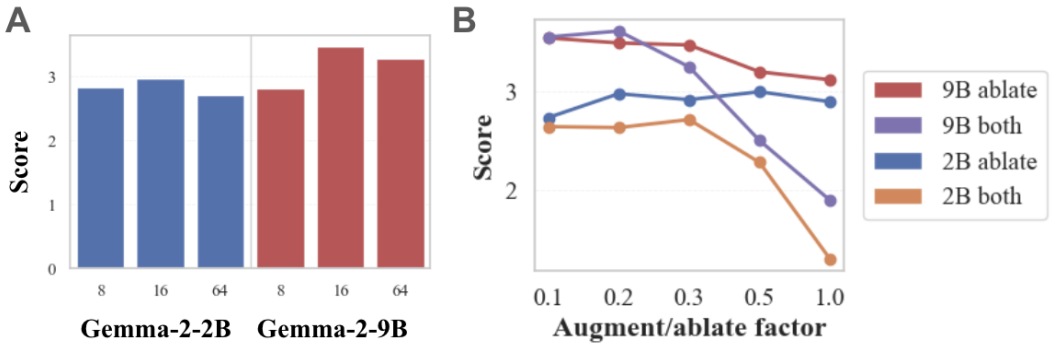

Figure 6: Hyperparameter sensitivity (MT-Bench with GPT-OSS-120B judge; higher is better). **A.** Score vs. k_{diff} (ablate only). **B.** Score vs. α (ablate only; augment+ablate). Performance peaks near k_{diff}=16 and $\alpha \approx 0.2$, and overly strong augment+ablate can degrade quality.

How many features to steer? Figure 6A shows the impact of modifying the parameter k_{diff}, the number of differential features to augment and/or ablate. We find that for both models, among the values surveyed, the best results are obtained for $k_{\text{diff}} = 16$.

How strong should steering be? In Figure 6B, we investigate the score obtained for various values of α, the factor that controls the steering strength. We observe a sharp decline in performance for high α values when both augmenting and ablating features, whereas there is only a modest decline when ablating only for Gemma-2-9B and little effect at all for Gemma-2-2B. This is related to the fact that we clamp ablated feature values from below by 0, whereas there is no cap for augmented features, allowing them to be overexpressed in the high-α regime. We find that $\alpha = 0.2$ yields the best results for Gemma-2-2B (ablate only) and Gemma-2-9B (augment and ablate) while maintaining high quality for other configurations; this motivates our choice of $\alpha = 0.2$ throughout.

5 Conclusion

We introduced DSPA, a prompt-conditional, inference-time preference-alignment method that steers decoding by modifying sparse SAE features via a conditional-difference map from preference data. Across MT-Bench and AlpacaEval on Gemma-2-2B/9B and Qwen3-8B, DSPA is competitive with strong inference-time baselines while preserving multiple-choice accuracy. DSPA remains robust under preference-data restriction and competitive with heavier pipelines while using far less alignment compute (up to $4.47\times$ fewer alignment-stage FLOPs than RAHF-SCIT). Because interventions occur in named SAE features, DSPA supports direct audits of what changes, and we find that preference gains are dominated by discourse and stylistic signals. Future work should test transfer to other safety and long-horizon tasks, and further explore de-mixing/whitening variants suggested by our theoretical analysis (Appendix A).

REFERENCES

Dana Arad, Aaron Mueller, and Yonatan Belinkov. Saes are good for steering – if you select the right features. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 10252–10270. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.519. URL http://dx.doi.org/10.18653/v1/2025.emnlp-main.519.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022. URL https://arxiv.org/abs/2204.05862.

Reza Bayat, Ali Rahimi-Kalahroudi, Mohammad Pezeshki, Sarath Chandar, and Pascal Vincent. Steering large language model activations in sparse spaces. *arXiv preprint arXiv:2503.00177*, 2025a.

Reza Bayat, Ali Rahimi-Kalahroudi, Mohammad Pezeshki, Sarath Chandar, and Pascal Vincent. Steering large language model activations in sparse spaces, 2025b. URL https://arxiv.org/abs/2503.00177.

Amrita Bhattacharjee, Shaona Ghosh, Traian Rebedea, and Christopher Parisien. Towards inference-time category-wise safety steering for large language models. *arXiv preprint arXiv:2410.01174*, 2024.

Abdelaziz Bounhar, Rania Hossam Elmohamady Elbadry, Hadi Abdine, Preslav Nakov, Michalis Vazirgiannis, and Guokan Shang. Yapo: Learnable sparse activation steering vectors for domain adaptation. *arXiv preprint arXiv:2601.08441*, 2026.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Bart Bussmann, Patrick Leask, and Neel Nanda. Batchtopk sparse autoencoders, 2024. URL https://arxiv.org/abs/2412.06410.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways, 2022. URL https://arxiv.org/abs/2204.02311.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with scaled ai feedback, 2024. URL https://arxiv.org/abs/2310.01377.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*, 2023.

Esin Durmus, Alex Tamkin, Jack Clark, Jerry Wei, Jonathan Marcus, Joshua Batson, Kunal Handa, Liane Lovitt, Meg Tong, Miles McCain, et al. Evaluating feature steering: A case study in mitigating social biases. *URL https://anthropic. com/research/evaluating-feature-steering*, 2024.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, et al. Toy models of superposition. *arXiv preprint arXiv:2209.10652*, 2022.

Jeremias Ferrao, Matthijs van der Lende, Ilija Lichkovski, and Clement Neo. The anatomy of alignment: Decomposing preference optimization by steering sparse features. *arXiv preprint arXiv:2509.12934*, 2025a.

Jeremias Ferrao, Matthijs van der Lende, Ilija Lichkovski, and Clement Neo. The anatomy of alignment: Decomposing preference optimization by steering sparse features, 2025b. URL https://arxiv.org/abs/2509.12934.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. *arXiv preprint arXiv:2406.04093*, 2024a.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024b. URL https://zenodo.org/records/12608602.

Lingkai Kong, Haorui Wang, Wenhao Mu, Yuanqi Du, Yuchen Zhuang, Yifei Zhou, Yue Song, Rongzhi Zhang, Kai Wang, and Chao Zhang. Aligning large language models with representation editing: A control perspective. *Advances in Neural Information Processing Systems*, 37:37356–37384, 2024.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval, 5 2023.

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2, 2024. URL https://arxiv.org/abs/2408.05147.

Wenhao Liu, Xiaohua Wang, Muling Wu, Tianlong Li, Changze Lv, Zixuan Ling, Zhu JianHao, Cenyuan Zhang, Xiaoqing Zheng, and Xuan-Jing Huang. Aligning large language models with human preferences through representation engineering. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10619–10638, 2024.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Van-Cuong Pham and Thien Huu Nguyen. Householder pseudo-rotation: A novel approach to activation editing in llms with direction-magnitude perspective. *arXiv preprint arXiv:2409.10053*, 2024.

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.

Senthooran Rajamanoharan, Tom Lieberum, Nicolas Sonnerat, Arthur Conmy, Vikrant Varma, János Kramár, and Neel Nanda. Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders. *arXiv preprint arXiv:2407.14435*, 2024.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15504–15522, 2024.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Yotam Wolf, Noam Wies, Dorin Shteyman, Binyamin Rothberg, Yoav Levine, and Amnon Shashua. Tradeoffs between alignment and helpfulness in language models with representation engineering. *arXiv preprint arXiv:2401.16332*, 2024.

Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. Axbench: Steering llms? even simple baselines outperform sparse autoencoders, 2025. URL https://arxiv.org/abs/2501.17148.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023. URL https://arxiv.org/abs/2306.05685.

Thomas P. Zollo, Andrew Wei Tung Siah, Naimeng Ye, Ang Li, and Hongseok Namkoong. Personalllm: Tailoring llms to individual preferences, 2025. URL https://arxiv.org/abs/2409.20296.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to ai transparency, 2025. URL https://arxiv.org/abs/2310.01405.

A Proofs and Additional Theory

A.1 Full Assumptions

Assumption A.1 (Shared-covariance mean shift, full statement)**.** There exist a positive semidefinite matrix $\Sigma \in \mathbb{R}^{d_{\text{SAE}} \times d_{\text{SAE}}}$, a scalar $c > 0$, and a prompt-dependent vector $\boldsymbol{\beta}(x) \in \mathbb{R}^{d_{\text{SAE}}}$ such that, conditional on x,

$$\tilde{\boldsymbol{\rho}}(y^+) \sim \big(\mu(x) + \tfrac{c}{2}\Sigma\boldsymbol{\beta}(x),\, \Sigma\big),$$
$$\tilde{\boldsymbol{\rho}}(y^-) \sim \big(\mu(x) - \tfrac{c}{2}\Sigma\boldsymbol{\beta}(x),\, \Sigma\big),$$

for some prompt-dependent mean $\mu(x)$. Equivalently, $\mathbb{E}[\Delta\tilde{\boldsymbol{\rho}} \mid x] = c\,\Sigma\boldsymbol{\beta}(x)$.

Assumption A.2 (Additive gating, full statement)**.** There exists a matrix $B \in \mathbb{R}^{d_{\text{SAE}} \times d_{\text{SAE}}}$ such that $\boldsymbol{\beta}(x) = B\, g(x)$. Let $M := \mathbb{E}[g(x)\, g(x)^\top] \in \mathbb{R}^{d_{\text{SAE}} \times d_{\text{SAE}}}$ denote the gate Gram matrix. Write the ith column of B as $\boldsymbol{\beta}^{(i)}$.

A.2 Proof of Theorem 3.3

Proof. By iterated expectation and Theorem 3.1,

$$\mathbf{A}^\top = \mathbb{E}\big[\Delta\tilde{\boldsymbol{\rho}}\, g(x)^\top\big] = \mathbb{E}\big[\mathbb{E}[\Delta\tilde{\boldsymbol{\rho}} \mid x]\, g(x)^\top\big] = c\,\Sigma\,\mathbb{E}\big[\boldsymbol{\beta}(x)\, g(x)^\top\big].$$

Using Theorem 3.2, $\boldsymbol{\beta}(x) = Bg(x)$, so $\mathbb{E}[\boldsymbol{\beta}(x)\, g(x)^\top] = \mathbb{E}[Bg(x)g(x)^\top] = B\,\mathbb{E}[g(x)g(x)^\top] = BM$, giving $\mathbf{A}^\top = c\,\Sigma\,B\,M$. Multiplying by Σ^{-1} on the left and M^{-1} on the right yields $\Sigma^{-1}\mathbf{A}^\top M^{-1} \propto B$. □

Interpretation. Equation $\mathbf{A}^\top = c\,\Sigma\,B\,M$ shows that rows of $\mathbf{A}$ generally mix multiple templates when gates co-activate: $\mathbf{A}_{i,:}$ is proportional to $\Sigma \sum_{i'} M_{ii'}\boldsymbol{\beta}^{(i')}$, not $\Sigma\boldsymbol{\beta}^{(i)}$ unless M is approximately diagonal.

A.3 Proof of Theorem 3.4

Proof. By Theorems 3.1 and 3.2, conditioning on $g_i = 1$:

$$\mathbb{E}[\Delta\tilde{\boldsymbol{\rho}} \mid g_i = 1] = c\,\Sigma\,\mathbb{E}[\boldsymbol{\beta}(x) \mid g_i = 1] = c\,\Sigma\,\mathbb{E}[Bg(x) \mid g_i = 1] = c\,\Sigma\left(\boldsymbol{\beta}^{(i)} + \sum_{i' \neq i} \pi_{i'|i}\boldsymbol{\beta}^{(i')}\right).$$

The norm bound follows from triangle inequality and submultiplicativity:

$$\|\Sigma \sum_{i' \neq i} \pi_{i'|i}\boldsymbol{\beta}^{(i')}\|_2 \leq \|\Sigma\|_{2\to 2} \sum_{i' \neq i} \pi_{i'|i}\|\boldsymbol{\beta}^{(i')}\|_2.$$

If additionally $\|\boldsymbol{\beta}^{(i)}\|_2 > 0$ and $\sum_{i' \neq i} \pi_{i'|i}\|\boldsymbol{\beta}^{(i')}\|_2 \leq \epsilon\|\boldsymbol{\beta}^{(i)}\|_2$, the relative error is bounded by $\epsilon \cdot \|\Sigma\|_{2\to 2}\|\boldsymbol{\beta}^{(i)}\|_2 / \|\Sigma\boldsymbol{\beta}^{(i)}\|_2$. □

A.4 DSPA Scoring and Optional De-mixing

At inference time, DSPA constructs an active set S_{prompt} (e.g., the top k_{prompt} indices by $\rho_i(x)$) and uses the truncated indicator $\hat{g}(x) := \mathbf{1}_{S_{\text{prompt}}} \in \{0,1\}^{d_{\text{SAE}}}$. The score is

$$\mathbf{s}(x) \;:=\; \mathbf{A}^\top \hat{g}(x) = c\,\Sigma\,B\,M\,\hat{g}(x),$$

where the ideal unmixed preference direction would be $c\,\Sigma\,B\,g(x)$. DSPA makes two approximations: (i) replacing $g(x)$ by a truncated indicator $\hat{g}(x)$, and (ii) ignoring the off-diagonal co-activation structure encoded in M. When $M \approx \mathrm{diag}(p)$ (weak gate dependence), $\mathbf{s}(x) \approx c\,\Sigma\,B\,\mathrm{diag}(p)\,\hat{g}(x)$.

Optional de-mixing on the active subspace. Although M is too large to invert globally, DSPA only uses a small active set. Let $S := S_{\text{prompt}}$ and $M_S := \mathbb{E}[g_S g_S^\top] \in \mathbb{R}^{|S|\times|S|}$. If M_S is well-conditioned, a de-mixed score is

$$\mathbf{s}_{\text{db}}(x) \;:=\; \mathbf{A}_{S,:}^\top\, M_S^{-1}\, g_S(x) \;\approx\; c\,\Sigma\,B_S\,g_S(x).$$

A.5 Optimality of Top-k Ablation

Assumption A.3 (Small ablation model in density space)**.** For a fixed prompt x, ablating a set $S \subset [d_{\text{SAE}}]$ of size k produces an expected change $\Delta\tilde{\boldsymbol{\rho}}(y) \approx -\delta\,\mathbf{1}_S$ for some fixed $\delta > 0$.

Theorem A.4 (Top-k ablation is optimal for linear utility)**.** *Under Theorem A.3 with linear utility $U(x, y) = \boldsymbol{\beta}(x)^\top \tilde{\boldsymbol{\rho}}(y)$, the expected utility improvement among all sets S with $|S| = k$ is maximized by choosing the k smallest coordinates of $\boldsymbol{\beta}(x)$.*

Proof. Under Theorem A.3, $\tilde{\boldsymbol{\rho}}$ shifts by $-\delta \mathbf{1}_S$, so $\mathbb{E}[U(x, y_{\text{edited}}) - U(x, y)] = -\delta \sum_{j \in S} \beta_j(x)$. With fixed k, this is maximized by selecting the k smallest $\beta_j(x)$. □

Relating $\mathbf{s}(x)$ to $\boldsymbol{\beta}(x)$. Under Theorem 3.1, the mean difference direction is $c\,\Sigma\boldsymbol{\beta}(x)$, so scoring coordinates by $\mathbf{s}(x)$ targets (a mixed approximation of) $\Sigma\boldsymbol{\beta}(x)$. If Σ is diagonal with nearly constant entries, rankings are preserved; otherwise, whitening by Σ^{-1} before selection may improve results.

A.6 Finite-Sample Concentration

Because $\Delta\tilde{\rho}_j \in [-1, 1]$ and $g_i \in \{0, 1\}$, each summand in $\hat{A}_{i,j}$ is bounded. Let $N_i := \sum_{k=1}^{N} g_i(x_k)$ be the number of training prompts with $g_i(x) = 1$.

Lemma A.5 (Row-wise concentration)**.** *Conditional on N_i, for any $\epsilon > 0$,*

$$\Pr\left(|\hat{A}_{i,j} - A_{i,j}| \geq \epsilon \mid N_i\right) \leq 2\exp\left(-\tfrac{1}{2} N_i \epsilon^2\right).$$

By a union bound, with probability at least $1 - \delta$,

$$\max_{j \in [d_{SAE}]} |\hat{A}_{i,j} - A_{i,j}| \leq \sqrt{\tfrac{2\log(2d_{SAE}/\delta)}{N_i}}.$$

Implication. Rare gates (small N_i) yield noisy conditional estimates, motivating (i) percentile thresholding to control gate frequency and (ii) conservative sparsification of $\mathbf{A}$ to remove entries dominated by estimation noise.

B Preference Feature Identification: Base vs. Fine-Tuned SAE

Let $\mathbf{A}^{\text{base}}$ denote the conditional-difference map (Section 3.1) computed using the base Gemma Scope SAEs, and $\mathbf{A}^{\text{custom}}$ the same map computed using the custom fine-tuned SAEs, in both cases using Gemma-2-9B with $\ell_{\text{input}} = 9, \ell_{\text{output}} = 39$. As an approximation for the features most likely to be augmented or ablated under DSPA, we consider the top $k_{\text{diff}} = 16$ output feature indices j for a given input feature i, sorting by $\mathbf{A}_{i,j}$ descending for augment features or ascending for ablate features. Denote by $\mathbf{A}_{i,(k)}$ the kth largest and by $\mathbf{A}_{i,(-k)}$ the kth smallest element of the row $\mathbf{A}_i$; taking the union overall all i yields

$$S^{\text{base}}_{\text{augment}} = \bigcup_{i \in [d_{\text{SAE}}]} \{j : \mathbf{A}^{\text{base}}_{i,j} \geq \mathbf{A}^{\text{base}}_{i,(16)}\},$$

$$S^{\text{base}}_{\text{ablate}} = \bigcup_{i \in [d_{\text{SAE}}]} \{j : \mathbf{A}^{\text{base}}_{i,j} \leq \mathbf{A}^{\text{base}}_{i,(-16)}\}$$

and likewise for $S^{\text{custom}}_{\text{augment}}$ and $S^{\text{custom}}_{\text{ablate}}$. We find that $|S^{\text{base}}_{\text{augment}}| = 183, |S^{\text{base}}_{\text{ablate}}| = 186, |S^{\text{custom}}_{\text{augment}}| = 76, |S^{\text{custom}}_{\text{ablate}}| = 83$. Intuitively, the set of features most likely to be modified for the fine-tuned SAE is several times smaller than that for the base SAE, indicating a stronger concentration of features along generalized preference axes in the fine-tuned case.

Interestingly, we find that many of the indices contained in these sets overlap between the base and fine-tuned SAE; concretely, $|S^{\text{base}}_{\text{augment}} \cap S^{\text{custom}}_{\text{augment}}| = 40, |S^{\text{base}}_{\text{ablate}} \cap S^{\text{custom}}_{\text{ablate}}| = 49$. Despite this, the generated interpretations do not always match; for example, feature 10776 is interpreted as *Questions or statements involving illegal or clearly harmful activities* for the fine-tuned SAE, and as *Hedging or clarification language expressing uncertainty or confusion* for the base SAE.

This small number of candidate features for the fine-tuned SAE is the grounds for our assertion in Section 3.1 that the matrix $\mathbf{A}^{\text{custom}}$ is sparse in practice. Let $\tau = \min\{|\mathbf{A}^{\text{custom}}_{i,j}| : \mathbf{A}^{\text{custom}}_{i,j} \geq \mathbf{A}^{\text{custom}}_{i,(16)} \text{ or } \mathbf{A}^{\text{custom}}_{i,j} \leq \mathbf{A}^{\text{custom}}_{i,(-16)}\}$. If we set entries of $\mathbf{A}^{\text{custom}}$ to 0 as long as $|\mathbf{A}^{\text{custom}}_{i,j}| < \tau$, then the number of nonzero entries is approximately $0.002 d^2_{\text{SAE}}$ and $\mathbf{A}^{\text{custom}}_{i,j} = 0$ for all $j \notin S^{\text{custom}}_{\text{augment}} \cup S^{\text{custom}}_{\text{ablate}}$. Generating responses to all first-turn MT-Bench questions and both ablating and augmenting features, we discover that the ablated features are a strict subset of $S^{\text{custom}}_{\text{ablate}}$ and the augmented features are a strict subset of $S^{\text{custom}}_{\text{augment}}$, validating this approximation.

C TRAINING COMPUTE COST COMPARISON: DSPA VS. RAHF-SCIT

We compare the cost of DSPA matrix construction against the two-stage RAHF-SCIT pipeline (Liu et al., 2024) under the assumptions used in our restricted-data analysis. Following standard dense-transformer FLOP accounting (Brown et al., 2020; Chowdhery et al., 2022), we approximate one forward token by $2P$ FLOPs and one training token (forward + backward) by $6P$ FLOPs for a dense P-parameter model, ignoring the smaller attention correction and implementation overhead. We take $P = 8 \times 10^9$ (an 8B-class model), and for DSPA use conservative estimates prompt/chosen/rejected lengths, setting $p = c = r = 1000$.

For DSPA, each preference triple requires three forward-only passes: one over the prompt, one over prompt+chosen, and one over prompt+rejected. Hence

$$F_{\text{DSPA}}(N) = 2P\,N(3p + c + r).$$

Under our length estimates, this becomes

$$F_{\text{DSPA}}(N) = 8 \times 10^{13} N.$$

For RAHF step 1, the analyzed configuration consumes $4N$ effective training examples (two epochs over duplicated chosen/rejected variants), each with two gradient-tracked passes at effective length $L_1 = 768$, following the hyperparameter configuration in (Liu et al., 2024). This gives

$$F_{\text{step1}}(N) = 48PNL_1 = 2.95 \times 10^{14} N.$$

For RAHF step 2, each example performs three no-grad forward passes and one gradient-tracked pass, for an effective cost of roughly $12P$ FLOPs/token. With effective optimizer-step batch $B = 64$, sequence length $L_2 = 512$, and $\texttt{max\_steps} = 0.02N$,

$$F_{\text{step2}}(N) = 12P(BL_2)(0.02N) = 6.29 \times 10^{13} N.$$

Therefore

$$F_{\text{RAHF}}(N) = F_{\text{step1}}(N) + F_{\text{step2}}(N) = 3.58 \times 10^{14} N,$$

so the modeled compute ratio is

$$\frac{F_{\text{RAHF}}(N)}{F_{\text{DSPA}}(N)} \approx 4.47.$$

These are model-FLOP estimates rather than measured hardware utilization, so the wall-clock ratio can be larger. In our Gemma-2-9B runs on a single Nvidia H200, DSPA matrix construction took 46 minutes with peak memory 33.1 GB, whereas the full RAHF pipeline took 8 hours 50 minutes with peak memory 140.8 GB. Thus, the observed wall-clock gap was $11.5\times$, substantially larger than the $4.47\times$ modeled FLOP ratio. This extra gap is consistent with implementation overheads that are not captured by the simple FLOP model, including fixed-length padding, gradient checkpointing, and the more complex multi-pass training loop used by RAHF.

D EXPERIMENTAL SETUP

D.1 EVALUATIONS

We obtain MT-Bench results using FastChat (`https://github.com/lm-sys/FastChat`), AlpacaEval results using the official repository (`https://github.com/tatsu-lab/alpaca_eval`), and MCQ benchmark results using the Language Model Evaluation Harness (`https://github.com/EleutherAI/lm-evaluation-harness`). Custom code is used for response and loglikelihood generation for each of these, while the provided code is used for evaluations. We use the common leaderboard settings and metrics for all MCQ benchmarks, as shown in Table 5.

Table 5: Multiple-choice benchmark settings.

Benchmark	Few-shot	Metric
MMLU	5	acc
Arc-Easy	25	acc_norm
TruthfulQA	0	mc2
HellaSwag	10	acc_norm
Winogrande	5	acc

D.2 BASELINES

Table 6 lists the hyperparameter configurations for our baselines. For single-layer interventions, we use the same output layer as with DSPA, namely $\ell_{\text{output}} = 24$ for Gemma-2-2B and $\ell_{\text{output}} = 39$ for Gemma-2-9B. For step 2 of RAHF, we run LoRA on layers $\{8, 10, 12, 14\}$ for Gemma-2-2B and $\{10, 12, 14, 16, 18\}$ for Gemma-2-9B, attempting to choose layers analogous to those used in the original paper for the given models. When performing our data ablation experiments, we scale down the max steps parameter on step 2 proportionately to avoid overfitting. Open-ended results are averaged over three generation seeds, and we did not conduct an extensive per-model hyperparameter sweep for training baselines (especially DPO); stronger baseline performance may be achievable with additional tuning.

Table 6: Baseline hyperparameters.

Baseline	Hyperparameter	Value
DPO	Epochs	1
DPO	Learning rate	5×10^{-6}
DPO	Beta	0.1
DPO	Optimizer	AdamW
RepE	Layer	ℓ_{output}
RepE	Directions	16
RepE	Scale	0.5
Static-SAE	Layer	ℓ_{output}
Static-SAE	Scale	0.5
RAHF	Mode	SCIT
RAHF (step 1)	Epochs	64
RAHF (step 1)	Learning rate	2×10^{-5}
RAHF (step 1)	Warmup ratio	0.1
RAHF (step 2)	Learning rate	3×10^{-4}
RAHF (step 2)	Max steps	450
RAHF (step 2)	LoRA rank	8
RAHF (step 2)	LoRA alpha	16
RAHF (step 2)	LoRA dropout	0.05
RAHF (step 2)	Alpha	5

E MT-BENCH CATEGORY BREAKDOWN

For more detailed inspection of the strengths and weakness of DSPA relative to our baselines, we provide a breakdown of MT-Bench scores by category across the surveyed methods, shown in Table 7. We find that DSPA is competitive across categories. Likely due to adjusting style and tone features, it scores the best on the Roleplay category for both models, while maintaining performance on more objective tasks such as Math and Coding. Indeed, it also performs the best of any method studied on Math for both models, although scores are low across the board there for the small models in question.

Table 7: MT-Bench scores by category.

		Coding	Extraction	Humanities	Math	Reasoning	Roleplay	STEM	Writing
Gemma-2-2B	DSPA (ours)	2.20	3.20	5.92	**2.40**	3.00	**6.60**	5.72	**6.05**
	DPO	**3.10**	4.38	**6.67**	2.00	**3.02**	4.95	5.65	5.09
	RepE	1.93	**4.97**	5.90	2.30	2.95	5.25	**5.96**	5.26
	Static-SAE	1.48	4.00	4.60	2.15	2.10	4.58	4.49	4.50
	Prompt Eng	2.05	3.20	5.90	**2.40**	2.15	5.03	5.72	4.05
	Base Model	2.30	3.80	5.90	2.20	2.90	4.05	5.92	4.17
Gemma-2-9B	DSPA (ours)	**3.55**	5.05	**6.78**	**2.35**	3.77	**5.55**	**7.17**	5.35
	DPO	3.30	**6.12**	5.67	2.25	**3.90**	5.10	6.67	**6.50**
	RepE	2.90	4.08	6.70	1.95	3.55	5.10	6.15	5.35
	Static-SAE	2.15	5.15	5.53	2.00	2.95	5.10	6.50	5.25
	Prompt Eng	3.10	5.10	6.55	1.85	3.27	4.68	6.53	5.70
	Base Model	3.33	3.90	5.86	1.95	3.45	5.12	6.55	4.92

F PERSONALIZATION EXPERIMENT

We also tested whether DSPA's interpretable latent interventions can support lightweight personalization. We use Gemma-2-9B with the same output SAE as in the main experiments. We curate 200 prompts containing varied content from the PersonalLLM dataset (Zollo et al., 2025), with an 80/20 train-test split. On the train set, we measure how ablating each candidate latent changes eight deterministic style axes relative to the base model, as shown in Table 10. We then construct persona-specific interventions either by selecting the top five latents with the highest estimated utility gain or by ablating all latents with positive estimated gain.

Evaluation is performed on the test set, where we report two complementary scoring protocols. First, we use a deterministic axis rubric: each response is scored on the eight axes and aggregated into a scalar utility under a fixed persona-specific weight vector. Second, we evaluate against the ten reward models released with PersonalLLM (Zollo et al., 2025), reporting mean judge z-scores (Table 9). For the latter method, we test two variants of the DSPA-based method, judge-axis and judge-contrast. Judge-axis DSPA selects latents using the deterministic eight-axis personalization rubric: we estimate how each latent shifts the hand-designed style axes, then rank latents by their predicted utility under a judge-specific preference profile derived from those axes. Judge-contrast DSPA instead selects latents directly from paired reward differences, favoring latents whose ablation most improves responses preferred by a given PersonalLLM judge over responses it disfavors, without going through the intermediate axis decomposition. We test against the following baselines:

1. **Instruction.** For each persona, we provide the model with a short natural-language description of that user's preferences and ask it to answer in a way that matches those preferences, without retrieving any example responses or modifying the model's activations.
2. **Few-shot ICL.** Following the most successful intervention from prior work (Zollo et al., 2025), we prepend one or five persona-matched example prompt-response pairs to the test prompt and then decode normally, without any latent intervention. The pairs are selected from the training set based on cosine similarity to the current prompt.

Across both protocols, latent ablation shows promise as a complementary personalization mechanism, though prompt-based baselines remain strong overall. On the deterministic rubric, DSPA alone (top 5) improves over the base model but trails both instruction-based and few-shot ICL methods; however, combining DSPA with 5-shot ICL yields the best results (3.173), suggesting that latent interventions and retrieval-based methods capture complementary signals. On the reward-model protocol, DSPA-based methods outperform instruction prompting but do not reach the performance of few-shot ICL. These results indicate that DSPA's stylistic interventions can augment other personalization strategies but are not yet a standalone substitute for retrieval-based approaches.

Table 10 shows the five single-latent ablations with the largest total absolute shift across the eight deterministic axes on the train split. The dominant patterns are broad discourse/style effects rather than clean one-axis controls: for example, latent 6345 most strongly shifts conciseness and proactivity, while latent 3491 strongly increases structure. This supports the view that DSPA personalization operates through reusable stylistic features.

Table 8: Personalization: deterministic utility (higher is better).

Method	Mean utility
Base model	2.400
DSPA (unconstrained)	2.431
DSPA (top 5)	2.539
Instruction	2.573
5-shot ICL	2.875
5-shot ICL + DSPA (top 5)	2.888
5-shot ICL + DSPA (unconstrained)	**3.173**

Table 9: PersonalLLM reward-model evaluation (mean judge z-score over 10 reward models; higher is better).

Method	Mean judge z-score
Instruction	-2.239
Judge-axis DSPA (top 5)	-2.128
Judge-contrast DSPA (top 5)	-2.061
1-shot ICL	-1.973
5-shot ICL	**-1.853**

G Feature Interpretation

G.1 Prompts

We use the following prompt to generate human-readable feature interpretations from `gpt-5-mini`:

```
System: You are an interpretability researcher focused on overall trends.
     Use the provided snippets as evidence, and tolerate some
    counterexamples. Only mark a feature as unreliable if you are
    genuinely unsure.

User: Latent {latent_idx} in an SAE. Provide a concise explanation (<=2
    sentences). If you are genuinely unsure, return reliable=false. In
    the snippets, tokens wrapped in [[double brackets]] are where the
    latent is active; use surrounding context to interpret what the
    latent represents. It is OK if the pattern has some exceptions.
Return JSON with keys: reliable (bool), explanation (string or null),
    evidence (string).
High-activation snippets: {snippets}
```

We again use `gpt-5-mini` with the following prompt to assign categories to each interpretation. Because the Safety and Intent categories had relatively few members, for clarity, we folded those into Content and Discourse respectively when reporting results in Section 4.3.

```
You are given a short description of a language feature (an SAE latent
    interpretation). Assign it to exactly one of the following categories
     based on its primary function. If multiple categories seem plausible,
     choose the most dominant one. Return only the category name, with no
     explanation.

Categories:
Safety - Language related to harm, illegality, sensitive activities,
    private data, refusal, caution, or policy-related framing (e.g.
    illegal activities, self-harm, drugs, personal data requests).
Discourse - Language that manages conversational flow or interaction
    rather than content (e.g. polite lead-ins, acknowledgements, hedging,
     topic shifts, explanation onsets).
Intent - Language that frames what kind of action or response is being
    requested or provided (e.g. asking for instructions, step-by-step
    guidance, advice framing, question vs answer signaling).
```

Table 10: Mean train-set axis shift Δ after ablating the five most influential latents (positive toward the left side of each axis).

Axis	6345	3491	1077	13585	4260
Conciseness vs. verbosity	-0.106	0.000	-0.044	0.019	0.044
Friendliness vs. professionalism	0.031	-0.025	0.019	0.031	0.031
Directness vs. diplomacy	-0.056	-0.081	-0.062	-0.100	-0.069
Structure vs. freeform	0.100	0.219	-0.050	0.050	0.000
Task focus vs. relational focus	-0.019	-0.019	-0.069	-0.038	-0.044
Proactive vs. minimal scope	-0.094	-0.006	-0.019	-0.069	-0.056
Certainty vs. caveating	-0.044	-0.075	-0.075	-0.056	-0.081
Creativity vs. literalism	0.006	-0.013	-0.050	0.013	-0.025

```
Structure - Tokens whose primary role is positional or structural rather
    than semantic (e.g. sentence-initial markers, clause boundaries, list
     markers, punctuation-adjacent or formatting tokens).
Grammatical - Closed-class or near-closed-class grammatical function
    words with minimal standalone meaning (e.g. articles, pronouns,
    auxiliaries, conjunctions, generic connectors).
Content - Topic-bearing or referential language with substantive semantic
     meaning (e.g. named entities, domains like politics or health,
    concrete objects or actions, lexical patterns like words starting
    with a specific prefix).
Feature description:
{description}
```

G.2 Ablate and Augment Set Features

Interpretations for the 50 ablate set and 50 augment set features identified in Section 4.3 are shown in Table 11 and Table 12, respectively.

H Disclosure on the use of artificial intelligence

Generative AI was used in this paper for assistance with writing some functions and methods and to flesh out details in the Appendix. The main body of the paper was written entirely by hand, and all code and writing produced by generative AI was thoroughly checked by the authors.

Table 11: Ablate-set features (50), with `gpt-5-mini` interpretations.

Feature	Interpretation
11569	Filler phrases used to smooth conversation and request clarification
10776	Questions or statements involving illegal or clearly harmful activities
6345	Polite opening phrases that frame what follows
11287	First-person statements expressing personal views or experiences
12550	Short connective phrases linking ideas across clauses
1077	Introductory phrases that set up a question or topic
1810	Brief acknowledgements and generic conversational replies
10995	Common opening phrases that ease into an explanation
4260	Discourse markers that signal a shift or continuation in thought
6167	Phrases that signal an explanation or reasoning is about to follow
8646	General noun phrases referring to people, roles, or categories
5681	Conjunctions commonly used at the start of multi-part statements
3107	Direct address phrases used at the beginning of a reply
14500	Very common function words and basic grammatical glue
13585	Requests asking for instructions or methods to achieve something
6782	Topic-introducing phrases that mark a new segment of thought
11172	Frequently used lead-in phrases starting an explanation
15059	Conversational opening phrases that introduce a question or new topic
3491	Common everyday words and short phrases spanning many unrelated topics
15061	Prefatory phrases that clarify, hedge, or gently frame what follows
13208	Short topical phrases that mark what is being asked about or explained
3869	Friendly, polite conversational lead-ins and transitional phrases
11799	Very common short words used to open questions or continue dialogue
5327	Requests asking for step-by-step instructions or practical guidance
5705	Small grammatical connectors linking clauses or introducing verb phrases
3605	Structural framing phrases that set up how information will be presented
15907	Compact noun or verb phrases carrying the main focus of a statement
13819	Discourse cue words that introduce, reference, or frame content
15659	Signals marking the start of a substantive explanatory continuation
3142	Food, cooking, and hosting-related language and recipe contexts
14408	Short function words and connective fragments near punctuation
12655	Generic introductory phrases that begin an explanation or transition
4456	Engaging, agreeable openings that signal readiness to explain or list
15107	General action- or advice-framing language introducing recommendations
15168	Very common function words with minimal standalone semantic content
6909	Clear explanatory or instructional language answering a question
11641	Sentence-initial grammatical tokens marking basic structure
8180	Clarifying lead-ins that refine or follow up
10118	Short grammatical tokens like pronouns and auxiliaries
5688	Sentence-opening filler with little consistent meaning
9580	Openings of short, direct questions
13493	Small connective words linking clauses
437	Question-framing cue words and auxiliaries
5811	High-level framing words introducing a topic or question
8003	Common pronouns and conjunctions as grammatical glue
2842	Pronouns and short connectors common in conversation
4035	Short determiners, pronouns, and auxiliaries
14791	Generic conversational utility phrases
9631	First-person statements expressing intentions or requests
11855	Connective function words continuing phrases

Table 12: Augment-set features (50), with `gpt-5-mini` interpretations.

Feature	Interpretation
141	Common function words and grammatical connectors without semantic content
9825	Clarification and intent-seeking phrases linking questions and follow-ups
10075	Topic-naming or subject-introducing words and phrases
1030	Short list items and connectors enumerating entities or phrases
11031	Affirmative explanatory openings that introduce a helpful continuation
8244	Contractions and possessive markers using apostrophes
15634	Sentence- or clause-initial tokens marking utterance beginnings
4891	Polite request openings introducing questions or actions
13301	Conversational scaffolding words framing dialogue or transitions
2969	Everyday life topics and general practical advice contexts
6268	Clause-initial function words like articles and auxiliaries
11779	Hedging or softening discourse openings before explanations
7569	Turn-initial or sentence-initial position markers
6509	Place names, locations, and location-focused references
6586	Very short replies handling brief or provocative inputs
2135	Prompts ending with a focused keyword or target phrase
9304	Response-leading fragments introducing explanations or suggestions
5314	Question phrasing and self-referential statements expressing personal perspective
5049	Auxiliary verbs and short grammatical continuations at clause boundaries
13656	Introductory interjections and brief discourse markers near turn starts
8172	Location and containment references like ”in”, ”in my area”, ”address”
4263	Key topical noun phrases defining the main semantic focus
12827	Frequent topical nouns and short subject phrases across contexts
866	Common connectors like ”or/of/the” in lists or alternatives
6143	Clause-beginning tokens and common connectors near sentence starts
1820	Very common function words and generic everyday connective tokens
8543	Conversational lead-ins introducing instructions or explanations
15344	High-frequency stopwords and short boundary tokens like ”the/from/to”
1477	Concise direct question forms often beginning with wh-words
7375	High-frequency grammatical connectors like ”are/this/the/but/is”
12366	Short glue phrases and function words near sentence or turn boundaries
8229	Initial tokens of named entities and multiword noun phrases
16271	Generic conversational openings and short transition phrases
14204	Salient topical keywords and notable content-bearing terms
5426	Informal conversational lead-ins, hedges, and smalltalk-style discourse markers
16033	Polite explanatory framing phrases introducing clarification or help
13130	Introductory phrases that open a new question or topic
6532	Very common function words and generic connective fillers
3594	Tokens marking clause boundaries and new structural segments
5416	Openings that begin or continue an explanatory reply
9670	Common connective words used during explanation or elaboration
13329	Direct question phrasing signaling a request for information
2610	Utterance-initial phrases starting a new topic or inquiry
613	Openings that introduce explanations, lists, or examples
7264	Words referencing the main topic or subject under discussion
15264	Short grammatical connectors like prepositions and determiners
3895	Utterance-initial tokens marking the very start of a clause
14998	Short function words anchoring clause starts or discourse pivots
2208	Sentence beginnings introducing concrete advice or factual responses
10754	Tokens marking the start of a new question or request