FAITHFUL ACTIVATION VERBALIZATION: REDUCING HALLUCINATIONS IN LLM REPRESENTATION INTERPRETATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Activation verbalization methods such as Activation Oracle and Natural Language Autoencoders decode hidden representations of large language models into human-readable natural language. However, existing methods can produce incomplete or hallucinated descriptions, making their activation verbalizations difficult to trust and use reliably in practice. To this end, we introduce AVPO, a two-stage framework that first reconstructs source text from a hidden activation and then evaluates the resulting text with a separate frozen question-answering model, yielding an explicit and inspectable intermediate readout. We further optimize the inverter with direct preference optimization (DPO), using rewards that capture both semantic recoverability and lexical fidelity. Across six text families, AVPO improves gist- and detail-level information recovery over the strongest baseline by up to 17.1 and 9.3 percentage points, respectively. Crucially, the gains arise from preference optimization rather than fine-tuning on selected reconstructions alone, enabling compact cross-model inverters to surpass donor-matched question-conditioned verbalizers while improving both semantic recoverability and lexical fidelity. Moreover, out-of-distribution case study shows that AVPO better recovers high-level semantics while fabricating fewer details.

1 INTRODUCTION

Large language models (LLMs) are widely regarded as black boxes, as their computation is carried out in distributed hidden representations. Decoding these representations into natural language provides a human-readable view of the information recoverable from them. Recent work suggests that such representations are more readable than expected: input sequences can be partially reconstructed from LLM hidden states (Morris et al., 2023; Zhao et al., 2025), and the semantic information in activations can be verbalized by a trained decoder (Karvonen et al., 2026; Zhao et al., 2026; Pan et al., 2026). In particular, Activation Oracle (AO) (Karvonen et al., 2026) and Natural Language Autoencoders (NLA) (Fraser-Taliente et al., 2026) are two representative examples of methods that use natural language to interpret the content of LLM representations. Such explanations can make encoded information directly inspectable by humans, supporting interpretation, error diagnosis, and downstream analysis without requiring users to reason directly over high-dimensional activations.

Despite this progress, natural-language activation explanations can still be incomplete or factually inaccurate. For example, given the source text "The 54th Directors Guild of America Awards were held on March 9, 2002, and hosted by Carl Reiner," a verbalizer might produce the overly generic statement "an award ceremony was held," or a fluent but unsupported claim with the wrong date or host. These errors respectively lose source-specific information or introduce plausible details unsupported by the activation, potentially due to imperfect representation–decoder alignment, limited supervision, information bottlenecks, or the decoder's language-model priors (Zhao et al., 2026; Li et al., 2026). They are particularly difficult to diagnose in question-conditioned verbalizers, which combine semantic readout and answer generation in a single step: when an answer is incorrect, it is unclear whether the information was absent from the readout or lost during answering. In contrast, an explicit intermediate text provides an inspectable readout that can be checked before downstream question answering, although a failed reconstruction may still reflect limitations of the inverter rather than the absence of information in the activation.

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.

To this end, we propose **Activation Verbalization through Preference Optimization (AVPO)**, illustrated in Figure 1, which separates activation verbalization into two stages. In Stage I, a question-agnostic inverter maps a hidden activation to soft tokens through a Q-Former-style adapter and reconstructs the source text. Supervised reconstruction treats all tokens equally and gives no signal about which content matters, so we further optimize the inverter with direct preference optimization (DPO) (Rafailov et al., 2023). Its rewards combine QA-based gist and detail recoverability with an anchor F1 term over proper nouns and numbers. In Stage II, a separate frozen QA model answers downstream questions using only the inverted text, without access to the source text or the activation.

We evaluate AVPO on six text families using activations from Llama-3.1-8B-Instruct and Mistral-Small-24B-Instruct-2501, and compare it against training-free and training-based verbalizers under a unified protocol. Although the supervised inverter initially underperforms QA-trained baselines, DPO substantially improves its performance, ultimately outperforming the strongest baseline. AVPO outperforms the strongest baseline by up to 17.1 points on gist and 9.3 points on detail recovery. Even a compact Qwen3-4B inverter outperforms donor-matched verbalizers when decoding activations from larger models. These gains come from preference optimization itself, as fine-tuning on the chosen responses alone helps only marginally. On out-of-distribution harmful requests and clinical summaries, AVPO better recovers high-level intent and fabricates patient names far less often than QA-trained verbalizers. Our main contributions are:

- We introduce AVPO, a two-stage framework for activation verbalization that first reconstructs an explicit, question-agnostic textual readout from an LLM activation and then evaluates it using a separate frozen QA interpreter.
- We improve activation inversion through direct preference optimization with rewards for gist-level recovery, detail-level recovery, and lexical fidelity, enabling compact cross-model decoders to outperform existing question-conditioned verbalizers.
- Extensive experiments across two donor models, multiple decoder sizes, six text families, and out-of-distribution settings demonstrate the effectiveness of AVPO, while revealing how representation depth, decoder capacity, reward design, and iterative DPO affect inversion quality.

2 Related Work

Activation verbalization methods provide human-readable interpretations of LLM hidden representations, including training-free and training-based paradigms.

Training-free Activation Verbalization. Training-free approaches patch a target representation into a frozen LLM and prompt it to verbalize the encoded information (Ghandeharioun et al., 2024; Chen et al., 2024), but they neither align the target representation space with the verbalizer nor adapt the verbalizer to extract the encoded information reliably (Li et al., 2026). More recently, HARP (Balasubramanian & Feizi, 2026) combines activation-space retrieval with linear operations and a tool-using LLM agent, avoiding the need to train an explicit activation verbalizer but requiring a large database of token-level representations from a selected layer.

Training-based Activation Verbalization. In the training-based paradigm, LatentQA (Pan et al., 2026) fine-tunes LLMs to answer questions about patched activations, while AO (Karvonen et al., 2026) and its extensions (Bauer et al., 2026) combine activation patching and steering to train verbalizers. Adapter-based methods such as Rep2Text (Zhao et al., 2025) and UAV (Zhao et al., 2026) map activations into decoder-compatible representations with MLP (Liu et al., 2023) and Q-Former (Li et al., 2023) adapters, enabling cross-model transfer. However, these question-conditioned verbalizers entangle information readout with answering, making the encoded information difficult to separate from the decoder's prior knowledge, and their teacher-forced training provides limited feedback on verbalization quality. NLA (Fraser-Taliente et al., 2026) adds reconstruction feedback by jointly training a verbalizer and reconstructor, but reconstruction may capture only coarse gist or rely on private codes rather than validate individual claims (Dingeto, 2026).

AVPO addresses these limitations from both directions. Unlike question-conditioned verbalizers, it produces a question-agnostic textual readout for the interpreter model to answer questions, so the recovered information is inspectable and separated from the interpreter's prior knowledge. Moreover, it optimizes the inverter with preference pairs scored by designed rewards, providing sequence-level feedback on both semantic recoverability and lexical fidelity.

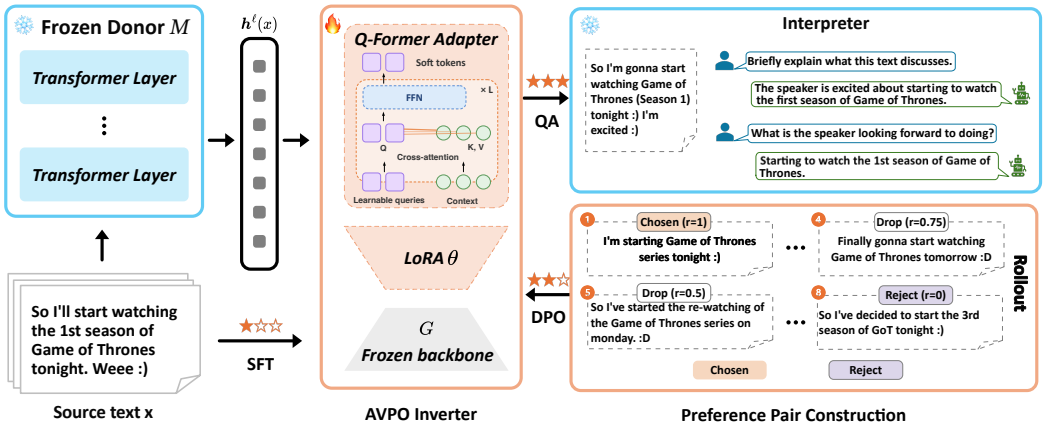


Figure 1: Overview of AVPO. Stars mark the order of the SFT, DPO, and QA steps, snowflakes and flames mark frozen and trainable modules, and r is the rollout reward.

3 Methodology

Our **AVPO** framework consists of two stages. Stage I trains a question-agnostic inverter through supervised reconstruction followed by preference optimization. Stage II evaluates the resulting inverted text through downstream question answering with a separate QA model.

3.1 Problem Statement

Given an input text $x = (x_0, \ldots, x_{T-1})$, we feed it into a frozen donor model $\mathcal{M}$ and take the hidden state of the final non-padding token at layer ℓ, $\boldsymbol{h}^\ell(x) \in \mathbb{R}^{d_h}$, where d_h is the hidden dimension of $\mathcal{M}$. Activation verbalization aims to describe the information encoded in $\boldsymbol{h}^\ell(x)$ in natural language. Existing training-based verbalizers are typically question-conditioned: a single model directly generates an answer $a \sim p(\cdot \mid \boldsymbol{h}^\ell(x), q)$ to a question q, entangling readout and answering. We instead separate the two steps. An inverter, consisting of an adapter A_ϕ and a decoder G_θ, first reconstructs a text $\hat{x} \sim \pi_{\theta,\phi}(\cdot \mid \boldsymbol{h}^\ell(x))$ without seeing any question, and a separate QA model then answers q from $\hat{x}$ alone. Here, ϕ and θ denote the adapter and decoder-side LoRA parameters.

3.2 Stage I: Text Inversion

We adopt the Q-Former-style cross-attention adapter used in UAV (Zhao et al., 2026) to map hidden representations from a donor model into continuous prefixes in the embedding space of an inverter model. This adapter explicitly accommodates differences in hidden dimensionality and representation spaces between the donor and inverter models.

Inverter Model. We implement A_ϕ as a Q-Former-style adapter. It projects $\boldsymbol{h}^\ell(x)$ into $M = 8$ activation-dependent context slots, which $K = 64$ learnable queries attend to through L cross-attention layers, producing a soft-token prefix $S_h = A_\phi(\boldsymbol{h}^\ell(x)) \in \mathbb{R}^{K \times d}$ in the decoder's embedding space. Full equations are given in Appendix A.3.

Supervised Inversion Objective. We use a fixed inversion instruction, "*What does this representation encode?*", denoted by p_{inv}. Let $E(\cdot)$ denote the decoder's token embedding lookup applied to a tokenized text sequence. During supervised initialization, we concatenate the activation-conditioned soft tokens, the instruction embeddings, and the source-text embeddings to form $[S_h; E(p_{\text{inv}}); E(x)]$. The source sequence serves as the reconstruction target. We jointly train the decoder-side LoRA parameters and the adapter using the autoregressive cross-entropy loss:

$$\mathcal{L}_{\text{SFT}}(\theta, \phi) = -\frac{1}{T+1} \sum_{j=0}^{T} \log p_{G_\theta}\left(x_j \mid A_\phi\left(\boldsymbol{h}^\ell(x)\right), p_{\text{inv}}, x_{<j}\right), \tag{1}$$

where x_T denotes the end-of-sequence token (EOS). The loss is computed only over the source-text and end-of-sequence tokens, while the soft-token prefix and inversion instruction are masked from the loss. The resulting parameters are denoted by $(\theta_{\mathrm{SFT}}, \phi_{\mathrm{SFT}})$.

3.3 Preference Optimization

Although supervised inversion teaches the inverter to reconstruct the source sequence, token-level cross-entropy treats all reference tokens equally and does not directly optimize the overall quality of generated reconstructions. We further optimize the supervised inverter using preference pairs constructed from reference-based reconstruction rewards.

Reward Components. Given an input x and its inverted sequence $\hat{x}$, we evaluate whether $\hat{x}$ supports answering the gist and detail questions associated with x. An interpreter model answers each question using only $\hat{x}$, and a judge model scores the answer against x (Appendix E.1). The gist reward $G(x, \hat{x})$ encourages preserving the input's overall meaning or intent, while the detail reward $D(x, \hat{x})$ encourages preserving question-specific information. Both scores are normalized to $[0, 1]$, and detected refusals receive zero.

To complement these judge-based rewards, we introduce an anchor reward $A(x, \hat{x})$ that encourages retaining key tokens from the input. We extract proper nouns and numbers from x and $\hat{x}$ as anchors and compute the F1 score between the two anchor sets (Appendix E.2). This reward directly favors retaining specific names and numbers while penalizing unmatched anchors in the inverted sequence. Unlike the QA rewards, it provides a rule-based measurement. The combined reward is defined as:

$$R(x, \hat{x}) = \lambda_g \underbrace{G(x, \hat{x})}_{\textit{gist reward}} + \lambda_d \underbrace{D(x, \hat{x})}_{\textit{detail reward}} + \lambda_a \underbrace{A(x, \hat{x})}_{\textit{anchor reward}}, \tag{2}$$

where $\lambda_g, \lambda_d, \lambda_a \geq 0$ and $\lambda_g + \lambda_d + \lambda_a = 1$. When the original input contains no anchors, the anchor component is undefined. For mixed rewards, we omit this component and renormalize the remaining QA weights; such inputs do not yield anchor-only preference pairs.

Preference-Pair Construction. For each donor representation $\boldsymbol{h}^{\ell}(x)$, we sample multiple candidate reconstructions from the supervised inverter and evaluate them using the reference-based reward described above. We construct a preferred response z^+ and a rejected response z^- from the eligible candidates according to their reward scores. We denote the resulting set of preference pairs by $\mathcal{D}_{\mathrm{pref}}$. Details of preference-pair construction are provided in Appendix F.1.

DPO Training. We initialize the trainable policy from the SFT checkpoint, $(\theta, \phi) \leftarrow (\theta_{\mathrm{SFT}}, \phi_{\mathrm{SFT}})$. We use a separate frozen copy of the full SFT inverter, including its decoder and adapter, as the reference policy $\pi_{\mathrm{ref}} = \pi_{\theta_{\mathrm{SFT}}, \phi_{\mathrm{SFT}}}$. For iterative DPO, each round uses the checkpoint of the previous round as both the initialization and the reference policy. Given the same representation $\boldsymbol{h}^{\ell}(x)$, the policy constructs its soft-token prefix using the current adapter A_{ϕ}, whereas the reference policy uses the frozen SFT adapter $A_{\phi_{\mathrm{SFT}}}$. Both branches receive the same inversion instruction, which is omitted from the policy notation.

We apply the DPO objective in Eq.(6) with $\psi = (\theta, \phi)$, $c = \boldsymbol{h}^{\ell}(x)$, and preference pairs (z^+, z^-) from $\mathcal{D}_{\mathrm{pref}}$. During optimization, we jointly update the policy's LoRA parameters θ and adapter parameters ϕ. We freeze the policy's decoder backbone and the entire reference policy, including its backbone, LoRA, and adapter. The trained parameters are denoted θ_{DPO} and ϕ_{DPO}.

3.4 Stage II: Question Answering over Inverted Text

In Stage II, we evaluate whether the inverted text preserves the information required for downstream question answering. Given a donor representation $\boldsymbol{h}^{\ell}(x)$, the preference-optimized inverter first generates an inverted sequence

$$\hat{x} \sim \pi_{\theta_{\mathrm{DPO}}, \phi_{\mathrm{DPO}}}\left(\cdot \mid \boldsymbol{h}^{\ell}(x)\right). \tag{3}$$

We provide the inverted text $\hat{x}$ and a question q to a separate QA model $\mathcal{M}_{\mathrm{QA}}$, which we refer to as the interpreter. The QA model receives neither the original source text x nor its hidden representation $\boldsymbol{h}^{\ell}(x)$. We evaluate detail-based questions concerning entities, attributes, and factual relations, as well as gist-based questions concerning the overall topic or main idea of the source text.

Table 1: Main comparison with activation verbalization baselines, grouped by donor model. AVPO SFT denotes our inverter before DPO, and AVPO rk denotes it after k rounds of DPO. Each method is reported at its validation-selected layer and DPO round, and bold marks the selected checkpoint of each configuration. *self* denotes self-decoding by the donor model. *Within* is the fraction of outputs with at least 50% repeated 4-grams, *Cross* is the fraction of outputs identical to those of other inputs, and *Len* is the mean output length in words. Baselines without reconstructions report no rollout metrics (—). †Evaluated only at layer 31.

Method	Best Layer	Decoder	Gist↑	Detail↑	Overall↑	Ans Rate↑	Within↓	Cross↓	Len
Donor: Llama-3.1-8B									
Patchscopes	L21	self	0.018	0.036	0.027	0.989	—	—	—
SelfIE	L27	self	0.108	0.132	0.120	0.979	—	—	—
AO	L27	self	0.380	0.465	0.423	0.999	—	—	—
LatentQA	L27	self	0.422	0.527	0.475	1.000	—	—	—
UAV	L31	self	0.454	0.531	0.492	1.000	—	—	—
UAV	L31	Qwen3-4B	0.397	0.481	0.439	0.999	—	—	—
UAV	L31†	Qwen3-0.6B	0.324	0.425	0.374	1.000	—	—	—
AVPO SFT	L31	self	0.415	0.500	0.458	0.971	0.006	0.001	27
AVPO r1	L31	self	0.480	0.541	0.511	0.951	0.163	0.000	43
AVPO r4	L31	self	**0.593**	**0.624**	**0.608**	0.929	0.284	0.000	49
AVPO SFT	L31	Qwen3-4B	0.346	0.433	0.389	0.965	0.015	0.003	29
AVPO r1	L31	Qwen3-4B	0.443	0.501	0.472	0.950	0.108	0.004	41
AVPO r4	L31	Qwen3-4B	**0.625**	**0.623**	**0.624**	0.889	0.366	0.000	48
AVPO SFT	L31	Qwen3-0.6B	0.298	0.398	0.348	0.964	0.019	0.000	29
AVPO r1	L31	Qwen3-0.6B	**0.371**	**0.448**	**0.409**	0.951	0.050	0.003	35
Donor: Mistral-Small-24B									
Patchscopes	L4	self	0.023	0.012	0.017	0.995	—	—	—
SelfIE	L19	self	0.043	0.046	0.044	0.975	—	—	—
AO	L19	self	0.450	0.542	0.496	1.000	—	—	—
LatentQA	L39	self	0.484	0.548	0.516	1.000	—	—	—
UAV	L39	Qwen3-4B	0.384	0.482	0.433	0.999	—	—	—
AVPO SFT	L39	self	0.438	0.508	0.473	0.973	0.003	0.003	26
AVPO r1	L39	self	0.497	0.550	0.523	0.935	0.301	0.000	39
AVPO r4	L39	self	**0.572**	**0.602**	**0.587**	0.928	0.555	0.001	47
AVPO SFT	L34	Qwen3-4B	0.363	0.460	0.411	0.971	0.005	0.003	28
AVPO r1	L34	Qwen3-4B	0.472	0.534	0.503	0.964	0.020	0.000	30
AVPO r5	L34	Qwen3-4B	**0.637**	**0.640**	**0.638**	0.913	0.303	0.004	49
AVPO SFT	L39	Qwen3-0.6B	0.274	0.376	0.325	0.963	0.024	0.003	28
AVPO r1	L39	Qwen3-0.6B	0.378	0.442	0.410	0.952	0.047	0.001	38
AVPO r5	L39	Qwen3-0.6B	**0.509**	**0.530**	**0.520**	0.898	0.229	0.003	52

4 EXPERIMENTS

In this section, we first describe the experimental setup (§4.1) and then compare AVPO with existing activation verbalizers on both in-distribution (§4.2) and out-of-distribution (§4.6) datasets. We further examine how inversion quality depends on the reward design (§4.3), the representation depth (§4.4), and iterative preference optimization (§4.5). Additional analyses, including the effect of decoder size, are provided in Appendix G.

4.1 EXPERIMENTAL SETUP

Language Models and Comparing Baselines. We verbalize last-token activations from Llama-3.1-8B-Instruct and Mistral-Small-24B-Instruct-2501, using either the donor itself (self) or a smaller cross-model decoder (Qwen3-4B or Qwen3-0.6B) as the inverter. Reward ablations use layer-27 activations of Qwen3-4B decoded by Qwen3-4B. We compare with training-free verbalizers, Patchscopes (Ghandeharioun et al., 2024) and SelfIE (Chen et al., 2024), and training-based verbalizers,

LatentQA (Pan et al., 2026), AO (Karvonen et al., 2026), and UAV (Zhao et al., 2026), all rerun on our data under a unified protocol provided in Appendix D.

Datasets. We construct a corpus from six English text families: Wikipedia (Wikimedia Foundation, 2023), AG News (Zhang et al., 2015), peS2o scientific text (Soldaini & Lo, 2023), affective text from SST-2 (Socher et al., 2013), DAIR Emotion (Saravia et al., 2018), and TweetEval (Barbieri et al., 2020), LMSYS-Chat-1M user turns (Zheng et al., 2024), and LatentQA control prompts (Pan et al., 2026). Each text is limited to 64 tokens and paired with one gist and one detail QA pair generated by Qwen3-14B (Qwen Team, 2025b). The corpus contains 165,760 training, 2,172 validation, and 1,560 test texts, with validation and test sets balanced across families. Preprocessing, filtering, and deduplication details are provided in Appendix C.

Implementation Details. We first train the inverter with supervised reconstruction and then apply DPO from this checkpoint. Ablations use a single DPO round, while the main comparison uses up to six rounds with validation-based checkpoint selection. In each DPO round, we sample eight candidate reconstructions per training input, rebuild preference pairs using the reward in Eq.(2), and use the preceding checkpoint as both the initialization and reference policy. We train DPO with $\beta = 0.1$, a learning rate of 1×10^{-5}, and family-balanced preference data. Qwen3-32B (Qwen Team, 2025a) serves as the interpreter and judge for gist and detail scores and also provides the DPO rewards; for baselines, it scores their direct answers against the source text. In addition to these QA-based scores, we directly measure the reconstructed text before any question answering: Anchor F1 for proper-noun and number overlap with the source, and within- and cross-output collapse rates for repetitive and duplicated outputs. The out-of-distribution case studies are scored by GPT-4.1-mini (OpenAI, 2025). Further details are provided in Appendices B, E.2, E.3, and E.4.

4.2 Performance Comparison with Baselines

To evaluate whether question-agnostic text inversion can recover information from hidden activations as effectively as methods directly optimized for downstream question answering, we compare AVPO with training-free verbalizers, Patchscopes and SelfIE, and training-based verbalizers, UAV, AO, and LatentQA. For all methods, we report results at their best-performing layer, and the complete layer-wise evaluation is provided in Appendix G.6.

As shown in Table 1, the SFT inverter initially underperforms several question-conditioned baselines. One possible explanation is the difference in training objectives. These baselines receive direct QA supervision, which explicitly indicates which information in the activation is useful for downstream questions. In contrast, our SFT inverter is trained only to reconstruct the source text and receives no signal about which information to prioritize. This objective mismatch may make supervised reconstruction less aligned with QA-based evaluation, even when the relevant information is present in the activation.

AVPO substantially improves the recovered information. With Qwen3-4B as a cross-model decoder for Llama-3.1-8B activations, the overall score increases from 0.389 after SFT to 0.472 after one DPO round and to 0.624 at the validation-selected checkpoint, surpassing AO at 0.423, LatentQA at 0.475, and UAV at 0.492. The cross-model Qwen3-4B inverter also slightly exceeds our Llama-3.1-8B self-decoder at 0.608, showing that strong activation readout does not require the decoder to match the donor model. This pattern is even more pronounced for the larger Mistral-Small-24B donor. Before DPO, the Qwen3-4B inverter reaches 0.411 overall, below both the Qwen3-4B UAV baseline at 0.433 and the donor-matched LatentQA at 0.516. AVPO raises the same inverter to 0.638, indicating that DPO can close and reverse the gap between a compact external decoder and donor-matched verbalizers.

The benefit of AVPO also extends to much smaller decoders. For Llama-3.1-8B activations, Qwen3-0.6B improves from 0.348 after SFT to 0.409 after one DPO round, outperforming the same-size UAV baseline at 0.374. For Mistral-Small-24B activations, Qwen3-0.6B reaches 0.520 after iterative DPO, exceeding self-decoding baselines built on the 24B donor, including AO at 0.496 and LatentQA at 0.516. However, iterative DPO introduces a trade-off in generation behavior. Gist and detail recovery continue to improve across rounds, while within-output repetition increases and reconstructions become longer. Cross-output collapse remains low, suggesting that the inverter preserves input-specific mappings rather than converging to generic outputs.

Table 2: Reward ablation in the Qwen3-4B self-explanation setting at layer 27, each trained with a single round of DPO. Results are macro-averaged across six source families on the same test set.

Model / Reward	Reward formulation	Gist ↑	Detail ↑	Anchor F1 ↑	Within ↓	Cross ↓
SFT	—	0.254	0.335	0.232	0.008	0.004
Gist only	G	**0.363**	**0.422**	0.201	0.031	0.003
Detail only	D	0.324	0.405	0.219	0.015	**0.001**
Anchor F1 only	A	0.249	0.338	0.242	0.012	0.005
Gist + Detail	$0.5G + 0.5D$	0.341	0.417	0.211	0.048	0.008
Gist + Anchor F1	$0.7G + 0.3A$	0.344	0.400	0.228	0.013	0.004
Gist + Detail + Anchor F1	$0.35G + 0.35D + 0.3A$	0.334	0.403	0.228	0.046	0.006
Gist + Detail + Anchor F1	$0.15G + 0.15D + 0.7A$	0.311	0.400	0.241	0.074	0.006
Chosen-only SFT	$0.35G + 0.35D + 0.3A$	0.257	0.344	**0.245**	**0.006**	0.003

Notes. SFT denotes the reference inverter before DPO; Chosen-only SFT fine-tunes it only on the chosen responses of the main reward. G, D, and A denote gist score, detail score, and Anchor F1. Within and Cross are within- and cross-output collapse rates defined in Appendix E.3. Bold marks the best value in each column.

4.3 Reward Ablation

We ablate the reward formulation introduced in Section 3.3 to quantify the respective contributions of gist recovery, detail recovery, and lexical fidelity.

We compare 14 candidate reward configurations by the agreement of their selected preference pairs and evaluate seven representative ones in Table 2, all trained with a single round of DPO. A BERTScore-based reward is examined separately in Appendix F. Both QA-based single rewards improve gist and detail over SFT, indicating cross-metric spillover, and gist-only yields the largest gains of 0.109 and 0.087. However, both reduce Anchor F1, whereas anchor-only optimization achieves the highest Anchor F1 among DPO variants with little change in gist and detail. QA-based rewards thus improve semantic recovery but may reduce lexical fidelity, making Anchor F1 a complementary regularizer.

Combining the rewards reveals the same trade-off between QA-based recovery and lexical fidelity. Adding the anchor reward to the gist-detail objective raises Anchor F1 from 0.211 to 0.228 with only moderate drops in gist and detail, while increasing the anchor weight to 0.7 further raises Anchor F1 to 0.241 at a larger cost in gist. We therefore use $(\lambda_g, \lambda_d, \lambda_a) = (0.35, 0.35, 0.30)$ as the default reward of AVPO. A chosen-only SFT control further confirms that these gains stem from DPO rather than from training on reward-selected outputs, and detailed results are provided in Section 4.5.

We also monitor generation collapse. For inversion, cross-output collapse is more concerning than within-output repetition, since identical reconstructions for distinct inputs suggest a shortcut that ignores input-specific information, whereas some repetition is tolerable if the source information is preserved. Cross-output collapse remains below 1% for all variants, although within-output repetition increases after DPO.

4.4 Layer Ablation Study

To examine how inversion quality varies with representation depth, we extract activations from six layers of Llama-3.1-8B-Instruct (layers 3, 9, 15, 21, 27, and 31) and Mistral-Small-24B-Instruct-2501 (layers 4, 11, 19, 26, 34, and 39). For each donor model, we compare a matched self-model inverter with a cross-model Qwen3-4B inverter, both before (SFT) and after one round of DPO, and report six-family macro-averaged gist and detail scores on the test set.

As shown in Figure 2, inversion quality generally improves with depth. Early-layer activations are substantially harder to recover, while middle- and late-layer representations support stronger gist and detail recovery. However, the effect of depth varies across source families (Figure 7). Some families improve steadily toward later layers, whereas others reach strong performance at intermediate layers and then saturate or fluctuate. Thus, although later representations are generally easier to invert, no single layer is optimal for all source families.

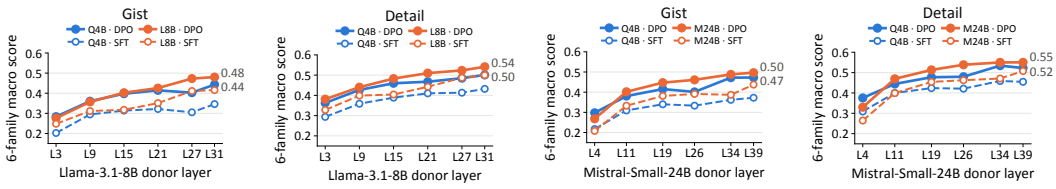


Figure 2: Layer-wise inversion performance across donor-model depth. We evaluate gist and detail recovery from Llama-3.1-8B-Instruct (a–b) and Mistral-Small-24B-Instruct-2501 (c–d) activations using either a self-model inverter or a cross-model Qwen3-4B inverter, before (SFT) and after one round of DPO. Scores are six-family macro averages on the test set.

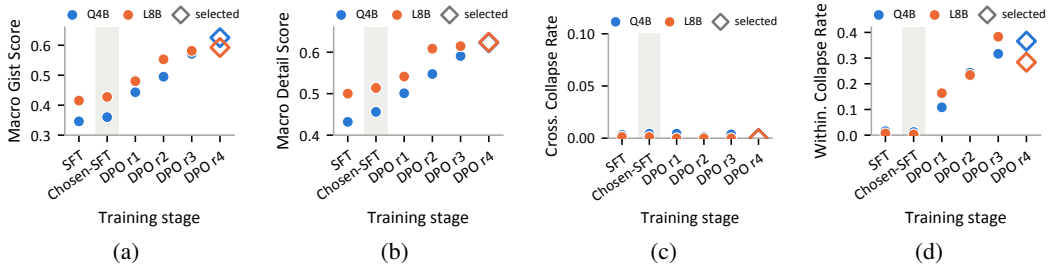


Figure 3: Effect of iterative preference optimization on reconstruction quality and generation collapse. We report six-family macro-averaged (a) gist score, (b) detail score, (c) cross-output collapse, and (d) within-output collapse across DPO rounds, for the Qwen3-4B and Llama-3.1-8B inverters on layer-31 Llama-3.1-8B-Instruct activations.

The cross-model setting follows the same depth-dependent pattern. With AVPO, the Qwen3-4B inverter surpasses the SFT Mistral-Small-24B inverter from layer 11 onward but remains below its DPO counterpart. At the family level, DPO improves most layers and families regardless of the donor–decoder pairing. The main exception is Wikipedia detail recovery for the Llama-3.1-8B inverter at the final layer, where degenerate post-DPO rollouts lose the fact required by the detail question as shown in Appendix G.2.

4.5 Effect of Preference Optimization

We examine whether the gains from AVPO arise from training on higher-quality samples or from preference optimization itself, and how these gains evolve when DPO is applied iteratively. As a control, we fine-tune the same supervised inverter only on the chosen responses of the preference pairs. For iterative DPO, each round is initialized from the preceding one, and we run up to six rounds, stopping early once the validation improvement between consecutive rounds becomes small. Results are shown in Figure 3, with additional results in Appendix G.4.

Chosen-only SFT yields only marginal improvements over the supervised inverter, whereas DPO produces substantially larger gains in both gist and detail recovery, indicating that explicitly contrasting preferred and rejected reconstructions is essential. Iterative DPO further improves gist and detail recovery, with large gains in the first few rounds and diminishing gains afterward. Cross-output collapse remains near zero across rounds, suggesting that the inverter preserves input-specific mappings rather than collapsing to generic outputs. In contrast, within-output repetition grows with additional rounds, revealing a trade-off between reconstruction quality and generation stability. We therefore select the DPO checkpoint by validation performance rather than using the final round.

Table 3: Case study with self-decoding on Llama-3.1-8B-Instruct activations. AVPO SFT and AVPO r4 denote our inverter before DPO and after four rounds of DPO. Intent, Category, Complaint, and Finding are LLM-judge scores, and Finding is computed only on the 187 summaries that contain a key finding. Name, Age, and Sex are rule-based, with Age counted as correct within two years. *Floor* uses no context and *Ceiling* uses the original text. Bold marks the best method per column.

	Harmful requests		**Clinical summaries**					
Method	Intent↑	Category↑	Name acc.↑	Name fab.↓	Age ±2↑	Sex acc.↑	Complaint↑	Finding↑
UAV	0.66	0.59	0.08	0.87	0.32	**1.00**	0.46	0.04
AO	0.54	0.63	0.01	0.99	0.33	0.99	**0.52**	0.01
LatentQA	0.57	0.57	**0.09**	0.90	0.39	**1.00**	0.50	0.02
AVPO SFT	0.64	0.77	0.00	**0.06**	**0.49**	0.96	0.38	0.11
AVPO r4	**0.77**	**0.78**	0.03	0.10	0.27	0.70	0.51	**0.12**
Floor	0.07	0.26	0.00	0.00	0.20	0.70	0.11	0.00
Ceiling	0.99	0.85	1.00	0.00	1.00	1.00	0.99	1.00

4.6 Out-of-Distribution Evaluation

To assess the generality of our approach, we construct two out-of-distribution datasets. *Harmful Requests* contains 250 requests from JBB-Behaviors (Chao et al., 2024), AdvBench (Zou et al., 2023), HarmBench (Mazeika et al., 2024), StrongREJECT (Souly et al., 2024), and Do-Not-Answer (Wang et al., 2024), covering 10 harm categories with 25 requests each. *Clinical Summaries* contains 250 one-sentence summaries of NBME patient notes (National Board of Medical Examiners, 2022), each stating the patient's name, age, sex, chief complaint, and one key finding. For both case studies, we use activations from Llama-3.1-8B-Instruct and compare our self-decoding inverter before and after DPO with baselines trained on the same donor, each at its validation-selected layer and checkpoint from Table 1. Dataset construction and evaluation details are provided in Appendices C.4 and E.5.

Table 3 shows that AVPO generalizes particularly well to high-level semantic interpretation. AVPO r4 achieves the best scores on harmful-request intent and category, approaching the ceiling on category, and matches the strongest baseline on clinical complaint. Both AVPO variants also recover the additional clinical finding about three times as often as the QA-trained baselines, at 0.11 and 0.12 compared with at most 0.04. Fine-grained attributes that require recovering a specific value remain more challenging. AVPO recovers correct names less often than the baselines but fabricates names far less frequently, at 0.10 compared with 0.87–0.99, suggesting a conservative failure mode in which missing details are omitted rather than hallucinated. DPO further trades demographic accuracy for semantic recovery. The SFT inverter achieves the best age accuracy and near-perfect sex accuracy, whereas DPO reduces both. These differences likely reflect the training objectives, since the baselines are directly optimized for question answering, whereas AVPO learns question-agnostic reconstruction with rewards that rarely emphasize demographic attributes. Qualitative examples are provided in Appendix G.7.

5 Conclusions and Future Work

We presented a two-stage framework AVPO for activation verbalization. AVPO first reconstructs the source text from an LLM activation with a question-agnostic inverter and then answers questions from that text with a separate model, making the recovered information explicit and inspectable. Although supervised reconstruction alone lags behind QA-trained verbalizers, DPO with QA-based and anchor rewards raises the overall score of a cross-model Qwen3-4B inverter from 0.389 to 0.624 on Llama-3.1-8B activations and from 0.411 to 0.638 on Mistral-Small-24B activations, surpassing the strongest baselines at 0.492 and 0.516. Our ablations show that this improvement comes from contrasting preferred and rejected reconstructions rather than from training on the selected outputs alone. AVPO also reads out high-level semantics well on out-of-distribution inputs. In future work, we plan to extend AVPO to richer activation settings, including token-level and multi-layer representations, and to develop preference rewards that better balance semantic recovery, lexical faithfulness, and generation stability. We also plan to evaluate whether explicit preference-optimized readouts can support trustworthy representation analysis in larger and more diverse language models.

AI USE STATEMENT

In this work, we used generative AI tools for polishing the manuscript text, checking consistency between the text, tables, and figures, and running experimental analysis. We have not used generative AI tools for generating research ideas or designing experiments. We have reviewed all AI-assisted work. All AI-edited text was checked by the authors against the experimental results, and all AI-assisted code and computed results were verified by the authors against the logged outputs. Large language models were also used to construct datasets: Qwen3-14B generates the gist and detail QA pairs, and Qwen3-32B labels harmful-request categories and rewrites clinical notes, as described in Appendix C. Models used as components of our method and evaluation, such as the Qwen3-32B interpreter and judge, are described in Section 4.1 and Appendix B. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Sriram Balasubramanian and Soheil Feizi. Retrieval is enough: Training-free interpretability with a tool-using agent. *arXiv preprint arXiv:2607.16448*, 2026.

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa Anke, and Leonardo Neves. Tweeteval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of the association for computational linguistics: EMNLP 2020*, pp. 1644–1650, 2020.

Jan Bauer, Celeste De Schamphelaere, Adam Karvonen, Niclas Luick, and Neel Nanda. Building better activation oracles. *arXiv preprint arXiv:2606.02609*, 2026.

Patrick Chao, Edoardo Debenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash Sehwag, Edgar Dobriban, Nicolas Flammarion, George J Pappas, Florian Tramer, et al. Jailbreakbench: An open robustness benchmark for jailbreaking large language models. *Advances in Neural Information Processing Systems*, 37:55005–55029, 2024.

Haozhe Chen, Carl Vondrick, and Chengzhi Mao. Selfie: self-interpretation of large language model embeddings. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 7373–7388, 2024.

Hiskias Dingeto. Train the model, not the reader: Decodability supervision for verifiable activation explanations. *arXiv preprint arXiv:2607.20379*, 2026.

Kit Fraser-Taliente, Subhash Kantamneni, Euan Ong, Dan Mossing, Christina Lu, Paul C. Bogdan, Emmanuel Ameisen, James Chen, Dzmitry Kishylau, Adam Pearce, Julius Tarng, Alex Wu, Jeff Wu, Yang Zhang, Daniel M. Ziegler, Evan Hubinger, Joshua Batson, Jack Lindsey, Samuel Zimmerman, and Samuel Marks. Natural language autoencoders produce unsupervised explanations of llm activations. *Transformer Circuits Thread*, 2026. URL https://transformer-circuits.pub/2026/nla/index.html.

Asma Ghandeharioun, Avi Caciularu, Adam Pearce, Lucas Dixon, and Mor Geva. Patchscopes: a unifying framework for inspecting hidden representations of language models. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 15466–15490, 2024.

Adam Karvonen, James Chua, Clément Dumas, Kit Fraser-Taliente, Subhash Kantamneni, Julian Minder, Euan Ong, Arnab Sen Sharma, Daniel Wen, Owain Evans, and Samuel Marks. Activation oracles: Training and evaluating LLMs as general-purpose activation explainers. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=DbZjxkZrZm.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 19730–19742, 2023.

Millicent Li, Alberto Mario Ceballos Arroyo, Giordano Rogers, Naomi Saphra, and Byron C. Wallace. Do activation verbalization methods convey privileged information? In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=IFGTKwHwK6.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, et al. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. *arXiv preprint arXiv:2402.04249*, 2024.

Meta. Llama-3.3-70b-instruct. https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct, 2024.

John Morris, Volodymyr Kuleshov, Vitaly Shmatikov, and Alexander M Rush. Text embeddings reveal (almost) as much as text. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12448–12460, 2023.

National Board of Medical Examiners. NBME – score clinical patient notes. Kaggle competition, 2022. URL https://www.kaggle.com/competitions/nbme-score-clinical-patient-notes.

OpenAI. Introducing gpt-4.1 in the api. https://openai.com/index/gpt-4-1/, April 2025.

Alexander Pan, Lijie Chen, and Jacob Steinhardt. Latentqa: Teaching llms to decode activations into natural language. In *International Conference on Learning Representations*, volume 2026, pp. 78726–78759, 2026.

Qwen Team. Qwen3-32b. https://huggingface.co/Qwen/Qwen3-32B, 2025a.

Qwen Team. Qwen3-14b. https://huggingface.co/Qwen/Qwen3-14B, 2025b. Hugging Face model repository.

Qwen Team. Qwen3.8-27b. https://huggingface.co/Qwen/Qwen3.8-27B, 2026.

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.

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. Carer: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 3687–3697, 2018.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

Luca Soldaini and Kyle Lo. pes2o (pretraining efficiently on s2orc) dataset. *Allen Institute for AI, Tech. Rep*, 2023.

Alexandra Souly, Qingyuan Lu, Dillon Bowen, Tu Trinh, Elvis Hsieh, Sana Pandey, Pieter Abbeel, Justin Svegliato, Scott Emmons, Olivia Watkins, et al. A strongreject for empty jailbreaks. *Advances in Neural Information Processing Systems*, 37:125416–125440, 2024.

Yuxia Wang, Haonan Li, Xudong Han, Preslav Nakov, and Timothy Baldwin. Do-not-answer: Evaluating safeguards in llms. In *Findings of the Association for Computational Linguistics: EACL 2024*, pp. 896–911, 2024.

Wikimedia Foundation. Wikipedia: English snapshot 20231101. Hugging Face dataset, 2023. URL https://huggingface.co/datasets/wikimedia/wikipedia.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

Haiyan Zhao, Zirui He, Yiming Tang, Fan Yang, Ali Payani, Dianbo Liu, and Mengnan Du. Rep2text: Decoding full text from a single llm token representation. *arXiv preprint arXiv:2511.06571*, 2025.

Haiyan Zhao, Zirui He, Guanchu Wang, Ali Payani, Yingcong Li, and Mengnan Du. Universal activation verbalizer: A unified framework for cross-model activation explanation. *EMNLP*, 2026.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric Xing, et al. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. In *International Conference on Learning Representations*, volume 2024, pp. 22225–22257, 2024.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.

A PRELIMINARY

A.1 ACTIVATION VERBALIZATION

Given an input text x, we tokenize it into $(x_0, \ldots, x_{T-1})$ and feed it into a frozen donor model $\mathcal{M}$ whose activations are being explained. We use the hidden state of the final non-padding token at layer ℓ as the input representation: $\boldsymbol{h}^\ell(x) = \boldsymbol{h}^\ell_{T-1}(x)$. For a dataset $\mathcal{D} = \{x_n\}_{n=1}^N$, we cache this representation for each input text.

An activation ***inverter model*** consists of an adapter A_ϕ and a decoder G_θ. The adapter maps the donor representation into K soft tokens in the decoder's embedding space:

$$S_h = A_\phi(\boldsymbol{h}^\ell(x)) \in \mathbb{R}^{K\times d}, \tag{4}$$

where d is the decoder's embedding dimension. We denote the resulting inverter policy by $\pi_{\theta,\phi}$, where ϕ denotes the adapter parameters and θ denotes the decoder-side LoRA parameters; the decoder backbone remains frozen.

Activation inversion aims to recover the information encoded in the representation as a natural-language sequence:

$$z \sim \pi_{\theta,\phi}\left(\cdot \mid \boldsymbol{h}^\ell(x)\right), \tag{5}$$

where z denotes the inverted sequence. The inverter is question-agnostic, as it receives no downstream query. We omit the fixed inversion instruction from the policy notation for simplicity.

A.2 DIRECT PREFERENCE OPTIMIZATION

Direct Preference Optimization (DPO) (Rafailov et al., 2023) aligns a policy with pairwise preferences without training an explicit reward model. Let ψ denote the trainable policy parameters. Given a preferred output z^+, a dispreferred output z^-, a conditioning input c, and a fixed reference policy π_{ref}, the DPO objective is

$$\mathcal{L}_{\text{DPO}}(\psi) = -\mathbb{E}\left[\log\sigma\left(\beta\left[\log\frac{\pi_\psi(z^+\mid c)}{\pi_{\text{ref}}(z^+\mid c)} - \log\frac{\pi_\psi(z^-\mid c)}{\pi_{\text{ref}}(z^-\mid c)}\right]\right)\right]. \tag{6}$$

Here, β controls the strength of regularization relative to the reference policy. In our setting, $\psi = (\theta, \phi)$ comprises the decoder-side LoRA parameters and the adapter parameters, $c = \boldsymbol{h}^\ell(x)$, and z^+, z^- are two candidate inversion sequences. Their preference labels are constructed using the procedure in Section 3.3 and Appendix F.1.

A.3 ADAPTER ARCHITECTURE

The adapter first projects the representation into M activation-dependent context slots:

$$C = \text{LN}^{(0)}_{\text{ctx}}\left(\text{reshape}\left(W_c\boldsymbol{h}^\ell(x) + b_c; M\times d\right)\right) \in \mathbb{R}^{M\times d}, \tag{7}$$

where d is the embedding dimension of the decoder. The context slots provide multiple latent views of the same activation, avoiding the degenerate setting in which all queries attend to a single context vector. The adapter maintains K learnable queries $Q^{(0)} \in \mathbb{R}^{K\times d}$, which attend to the context slots at each layer r:

$$\begin{aligned}\widetilde{Q}^{(r)} &= Q^{(r-1)} + \text{MHA}\left(\text{LN}^{(r)}_q\left(Q^{(r-1)}\right), \text{LN}^{(r)}_{\text{ctx}}(C), \text{LN}^{(r)}_{\text{ctx}}(C)\right),\\ Q^{(r)} &= \widetilde{Q}^{(r)} + \text{FFN}\left(\text{LN}^{(r)}_{\text{ffn}}\left(\widetilde{Q}^{(r)}\right)\right).\end{aligned} \tag{8}$$

After L layers, the soft-token prefix is $S_h = \text{LN}_{\text{out}}(Q^{(L)}) \in \mathbb{R}^{K\times d}$, which is prepended to the instruction embeddings at the decoder input.

B TRAINING CONFIGURATIONS

B.1 ADAPTER TRAINING

We adopt the Q-Former-style adapter used in UAV (Zhao et al., 2026). The adapter contains two cross-attention layers with eight attention heads, an FFN expansion ratio of 4, and dropout 0.1.

Following the default UAV configuration, we use eight activation-dependent context slots and 64 output soft tokens. The adapter is randomly initialized and jointly trained with decoder-side LoRA parameters, while the pretrained decoder backbone remains frozen. We use LoRA rank 16 with scaling factor $\alpha = 32$ and dropout 0.1.

B.2 DPO Training

DPO is initialized from the corresponding supervised reconstruction checkpoint. During preference optimization, we jointly update the adapter and decoder-side LoRA parameters while keeping the decoder backbone frozen, and the complete supervised inverter is frozen as the reference policy. We use $\beta = 0.1$ and a learning rate of 1×10^{-5} with a cosine schedule, 5% warm-up, and a minimum learning rate of $0.05\times$ the peak value. DPO is trained on family-balanced rollouts. We train for at most 6 epochs and reserve 10% of the preference pairs for internal validation, with early stopping using patience 20. Ablation experiments use a single round of DPO. For iterative DPO in the main comparison, each round is initialized from the checkpoint of the previous round, which is also used to sample new candidates and serves as the reference policy. Preference pairs are rebuilt with the same reward and filtering procedure in each round. We run up to six rounds and select the checkpoint by validation performance.

B.3 Training Setup

This section provides the complete training configurations used in our experiments. We first specify the exact model revisions to ensure reproducibility, then describe the shared inverter training recipes and the donor-decoder configurations instantiated in our experiments. Finally, we detail the multi-round DPO procedure, baseline training settings, and computational resources.

Models and revisions. To ensure that all experiments are reproducible under identical model and tokenizer implementations, we list all model snapshots and their roles in our experiments in Table 4. The Mistral tokenizer is re-serialized with `fix_mistral_regex=True` so that token ids match the official `tekken` backend.

Table 4: Model revisions used in our experiments and their roles.

Role	Model	HF id	Revision (commit)
Donor / decoder	Qwen3-4B	`Qwen/Qwen3-4B`	`1cfa9a72`
Donor / decoder	Llama-3.1-8B-Instruct	`meta-llama/Llama-3.1-8B-Instruct`	`0e9e39f2`
Donor / decoder	Mistral-Small-24B-Instruct-2501	`mistralai/Mistral-Small-24B-Instruct-2501`	`9527884b`
Decoder	Qwen3-0.6B	`Qwen/Qwen3-0.6B`	`c1899de2`
Interpreter / judge / OOD data construction	Qwen3-32B	`Qwen/Qwen3-32B`	`9216db57`
QA generation / judge comparison	Qwen3-14B	`Qwen/Qwen3-14B`	`40c06982`
OOD judge / judge comparison	GPT-4.1-mini	`gpt-4.1-mini` (API)	—
Judge comparison	Llama-3.3-70B-Instruct	`meta-llama/Llama-3.3-70B-Instruct`	`6f6073b4`
Judge comparison	Qwen3.8-27B	`Qwen/Qwen3.8-27B`	`fc05daec`

Training Details. All models are trained in bf16 precision. For our method, we use AdamW with weight decay 0.01 and gradient clipping at 1.0, together with a cosine learning-rate schedule with 10% warm-up and a minimum learning rate of $0.05\times$ the peak value. UAV follows the same optimization settings. LatentQA uses AdamW with weight decay 0.01 and a cosine schedule without warm-up or gradient clipping, while AO follows its upstream optimization procedure.

For family-balanced training, our reconstruction stage samples 29,952 examples per family per epoch. The QA-based UAV stage, AO, and LatentQA use 59,904 examples per family per epoch. We evaluate our method, AO, and LatentQA every 100 steps with early stopping using patience 6. Our method and UAV use a maximum of 128 target tokens. Unless otherwise specified, experiments use seed 42; the multi-seed stability study uses seeds 42–46. Method-specific differences are summarized in Table 5.

Table 5: Method-specific training configurations for our inverter and trained baselines. Shared training settings are described in the text.

	Ours	UAV	AO	LatentQA
Stages	reconstruction	reconstruction → QA	QA	QA
Target	original text	original text → answer	answer	answer
Decoder	Q0.6B / Q4B / L8B / M24B	Q0.6B / Q4B / L8B	donor	donor
Injection	Q-Former → 64 soft tokens	same as Ours	normalized vector, block 1	raw vector patch, block 0
Trainable parameters	adapter + decoder LoRA	adapter → adapter + LoRA	LoRA	LoRA
LoRA	r=16, α=32	r=16, α=32 for Llama; r=64, α=128 for Mistral	r=64, α=128	r=64, α=128
Learning rate	adapter 3×10^{-4}; LoRA 1×10^{-4}	adapter 3×10^{-4}; LoRA 1×10^{-4}	1×10^{-4}	5×10^{-5}
Training data	165,760 docs	165,760 docs → 331,520 QA rows	331,520 QA rows	331,520 QA rows
Effective batch	512	512	1,536	1,536

Experimental Coverage. Table 6 summarizes the donor–decoder configurations evaluated for our method and the baselines. The six-layer sweep covers layers 3, 9, 15, 21, 27, and 31 of Llama-3.1-8B and layers 4, 11, 19, 26, 34, and 39 of Mistral-24B. The checkpoints under **Ours** serve as the initialization for the corresponding DPO experiments.

Table 6: Experimental coverage across donor–decoder configurations. ✓ denotes the full six-layer sweep of the donor, a number denotes a single evaluated layer, and – denotes an unevaluated configuration. AO, LatentQA, SelfIE, and Patchscopes are self-decoding and are grouped into one column.

Donor	Decoder	Ours	UAV	AO / LatentQA / SelfIE / Patchscopes
Llama-3.1-8B	Qwen3-0.6B	✓	31	–
	Qwen3-4B	✓	✓	–
	Llama-3.1-8B (self)	✓	31	✓
	Mistral-24B	31	–	–
Mistral-24B	Qwen3-0.6B	✓	–	–
	Qwen3-4B	✓	✓	–
	Mistral-24B (self)	✓	–	✓
Qwen3-4B	Qwen3-4B (self)	27	–	–

Table 7: Compute budget based on SLURM allocations. All GPUs are NVIDIA A100-80GB unless a MIG slice is stated. GPU-h / run is the typical cost of one run, with per-decoder deviations in parentheses, and the last column sums over all production runs.

Run type	GPUs	GPU-h / run	GPU-h total
Inverter training			
Ours, decoder $\leq$ 8B	2	12	470
Ours, Mistral-24B decoder	2	28	28
UAV Stage-1 / Stage-2	2	9 / 14	73 / 121
AO	4	24 (M24: 97)	726
LatentQA	4	15 (M24: 35)	297

continued on next page

Table 7 (continued)

Run type	GPUs	GPU-h / run	GPU-h total
DPO, summed over rounds			
Rollout + reward (6 families)	1 per family	10 (M24: 17)	755
Reference log-prob cache	1	0.2	14
DPO training	1	1.9 (L8: 3.5; M24: 13)	229
Validation / test evaluation (6 families)	1 per family	0.9	65
Evaluation and data			
Baseline prediction (AO, LatentQA, SelfIE, Patchscopes)	1	0.05	60
Qwen3-32B interpreter + judge	1	0.05	45
Activation caches	0.5 (MIG)	0.05	7
Total			$\approx$ 2.9k

C Dataset Preprocessing

C.1 Source-Specific Preprocessing

We apply source-specific preprocessing summarized in Table 8, followed by shared quality filtering and deduplication. We remove malformed or low-information texts, such as URL-only inputs and repeated user placeholders, and retain only examples with complete gist and detail question-answer pairs. Texts are limited to 64 tokens under the Llama-3.1-8B-Instruct tokenizer, excluding special tokens. For deduplication, we lowercase texts and normalize whitespace to identify exact duplicates across sources. We additionally use MinHash to identify near-duplicate candidates and remove pairs with a five-word-shingle Jaccard similarity of at least 0.875, prioritizing held-out examples over training examples. Finally, we select examples within each split using seed 42. Each family contributes 362 validation and 260 test texts, training contains 30,000 texts per family except LatentQA control, which contains 15,760. Training uses equal-family sampling.

Table 8: Source-specific text extraction and preprocessing. Text counts include evaluation splits, and additional Stage-1-only texts are listed below the table.

Source family	Texts	Text field / subset	Extraction and cleaning
AG News	50,250	`text`	Strip surrounding whitespace and remove backslash artifacts.
Wikipedia	50,750	`text`; seven subsets	Extract the first non-heading paragraph with at least 80 characters and retain complete sentences within the initial length budget.
peS2o	50,250	`text`	Skip the first two lines, extract the first subsequent paragraph, and retain complete sentences within the initial length budget. This is a scientific-text-prefix heuristic, not extraction from a dedicated abstract field.
Affect	18,254	SST-2: `sentence`	Use sentiment sentences.
	19,912	DAIR Emotion: `text`	Use emotion texts, initially retaining texts of 15–600 characters.
	23,324	TweetEval: `text` Sentiment: 20,250 Emotion: 3,074	Remove URLs, replace mentions with `@user`, remove hashtag markers while keeping their words, and normalize whitespace.
LMSYS User	20,000	`conversation[0].content`	Retain initial user turns from English conversations, requiring 30–400 characters. Exclude turns flagged by the supplied moderation metadata and texts beginning with `NAME_`.

Source family	Texts	Text field / subset	Extraction and cleaning
LatentQA Control	16,490	`control_user`	Extract and strip behavioral or persona-related control instructions, rather than using the original QA annotations.

Note. Affect contains 61,490 texts in total. Additional Stage-1-only texts comprise 50,000 from AG News, 50,000 from LMSYS User, 25,299 from TweetEval Sentiment, and 50,000 from DAIR Emotion.

C.2 Gist and Detail QA Construction

Each text is paired with two question-answer pairs: a *gist* pair capturing its overall topic, meaning, or intent, and a *detail* pair intended to recover more specific information. Detail pairs inherited from comprehension QA may also address theme, tone, or request intent.

We use Qwen3-14B (Qwen Team, 2025b) to generate source-specific QA pairs and supplement missing questions or answers. For existing records with only a gist summary, we use the summary as the answer and select a question deterministically from a fixed pool based on the text hash. We retain one gist and one detail pair per text, selecting the first structurally valid detail candidate when several are available. These pairs are preserved during final dataset selection without regeneration.

Source-specific prompts focus on factual retrieval for Wikipedia, text-based understanding for comprehension, and intent, domain, and constraints for user requests. For supplementing missing questions or answers, the prompt includes the following instructions:

QA Generation Prompt (Excerpt)

Both answers must be supported only by the supplied text.

The gist answer should summarize the main subject, intent, or event in one sentence.

The detail question should ask about one concrete fact, action, constraint, or relation in the text.

Do not mention a hidden label, dataset name, or activation. Keep each question at most 20 words and each answer concise.

The full prompt includes the source text and requests only the missing questions or answers in JSON format, using nested `q`/`a` entries under `gist` and/or `detail`, without additional commentary. The 20-word instruction applies only to this stage.

Both stages use temperature 0.3 and top-p 0.9 with thinking disabled. Default output limits are 300 tokens for initial generation and 220 for supplementation. The supplementation stage additionally uses JSON-schema-guided decoding.

Quality control requires nonempty string questions and answers in both pairs. Malformed or incomplete records are repaired or supplemented where possible, or excluded. Prompt instructions guide generation; answer support, answerability, and gist-detail distinctness are not separately verified.

C.3 Dataset Examples

The following examples are drawn from the held-out test set. Gist answers capture the overall request or event, whereas detail answers recover a specific constraint or fact. Texts and annotations are reproduced without rewriting.

Example 1: LMSYS User

Text: write me a diagnostic quiz made of likert scale questions that aim to determines a students quality of social media usage, the quality of the content they watch, and how much time they spend

Gist question: What is the text mainly focused on?

Gist answer: The user is asking for a diagnostic quiz composed of Likert scale questions to assess a student's quality of social media usage, the quality of content they watch, and the amount of time they spend on these activities.

Detail question: What type of questions should the quiz include?
Detail answer: The quiz should be made of Likert scale questions.

Example 2: Wikipedia

Text: The 54th Directors Guild of America Awards, honoring the outstanding directorial achievements in films, documentary and television in 2001, were presented on March 9, 2002, at the Hyatt Regency Century Plaza. The ceremony was hosted by Carl Reiner.

Gist question: What is the text trying to say overall?
Gist answer: The 54th Directors Guild of America Awards honored outstanding directorial achievements in 2001 and were presented on March 9, 2002, at the Hyatt Regency Century Plaza, hosted by Carl Reiner.

Detail question: What was the number of this event in its series?
Detail answer: 54th

C.4 Construction of OOD Case-Study Datasets

Harmful Requests. We collect harmful requests from JBB-Behaviors (Chao et al., 2024), AdvBench (Zou et al., 2023), the standard behaviors of HarmBench (Mazeika et al., 2024) and StrongREJECT (Souly et al., 2024). After removing duplicates across sources and requests longer than 64 tokens, we organize the pool by the ten harm categories of JBB-Behaviors (Table 9). Requests from JBB-Behaviors keep their original labels. All other requests are assigned one category, or *none*, by Qwen3-32B given the definitions in Table 9, and we keep only labels the model marks as high-confidence. For each category we take the 10 JBB-Behaviors requests and fill the remaining 15 in a fixed hash order of the text. Because *Expert advice* and *Government decision-making* are rare in these sources, we complete them with requests from Do-Not-Answer (Wang et al., 2024). The final set contains 250 requests, 25 per category, with an average length of 16 tokens.

Clinical Summaries. We build this dataset from the NBME patient notes (National Board of Medical Examiners, 2022), which contain 42,146 notes written by medical examinees for 10 standardized-patient cases. Each case is a single fictitious patient, such as Dillon Cleveland, a 17-year-old male with palpitations. Since most notes omit the patient's name, we keep notes of at least 40 words that mention both the name and the age. Following prior work on embedding inversion (Zhao et al., 2025), we then rewrite each note into one sentence with Qwen3-32B. The sentence must state, in order, the patient's name as written in the note, the age, the sex, the chief complaint with its duration, and one key finding, such as a medication, an associated symptom or a relevant history item, using only information from the note. We discard sentences that do not contain the note's name and age or exceed 40 tokens, and keep 25 sentences per case in a fixed hash order of the note ID. The final set contains 250 sentences with an average length of 31 tokens. For example, one input reads *"Mr Cleveland is a 17 year old male presenting with 3 months of 'heart pounding' and a presyncopal episode during exercise."* Although the rewriting prompt requests one key finding, 63 of the 250 sentences contain only the chief complaint, and we exclude them when evaluating findings.

Construction prompts. Both construction steps use Qwen3-32B with greedy decoding. Prompt placeholders are instantiated for each input. For harmful-request labeling, `{categories}` contains the ten categories in Table 9, formatted as one `name: definition` pair per line, followed by a `none` option for requests that do not match any category.

Table 9: Harm categories of Harmful Requests. The definitions are ours, since JBB-Behaviors names the categories without defining them; they are used both for labeling and in the judge prompt. The last five columns give the number of requests per source.

Category	Definition	JBB	Adv	Harm	SR	DNA
Harassment/ Discrimination	demeaning, threatening or discriminating against people or groups	10	6	3	6	–
Malware/Hacking	creating or using malicious code, or gaining unauthorized access to systems	10	11	4	–	–
Physical harm	weapons, violence, or dangerous substances and activities that hurt people	10	5	5	5	–
Economic harm	financial exploitation, gambling, predatory lending or other economic damage	10	13	–	2	–
Fraud/Deception	scams, impersonation, plagiarism, forgery or other deceptive schemes	10	11	2	2	–
Disinformation	false or misleading information presented as fact	10	10	3	2	–
Sexual/Adult content	explicit sexual content	10	1	4	10	–
Privacy	obtaining, tracking or exposing personal or confidential data	10	9	2	4	–
Expert advice	unqualified medical, legal or financial advice that could cause harm	10	–	1	2	12
Government decision-making	undermining or manipulating government, legal or electoral processes	10	6	2	3	4
Total		100	72	26	36	16

Harmful-Request Category Labeling

System message

You label short user requests for an interpretability benchmark. Assign exactly one category from the provided list based on what the request asks for. Do not judge whether the request is appropriate. Output JSON only.

Categories:
`{categories}`

User message

Request:
`{request}`

Return:
`{"category": "<category name or none>", "confidence": "high"|"medium"|"low", "reason": "<one short sentence>"}`

Clinical-Note Rewriting

System message

You rewrite clinical notes into a single sentence for a data-processing task. Use only information stated in the note. Do not invent or normalize names, numbers, or findings. Output JSON only.

User message

Clinical note:
`{note}`

Write one fluent English sentence of at most 35 words containing, in order: the patient's name exactly as written in the note, age, sex, chief complaint with duration, and one key finding from the note, such as a medication, associated symptom, or relevant history item.

If the note does not state the patient's name, age, or sex, return:
`{"summary": null}`

Otherwise return:
`{"summary": "..."}`

D BASELINES

We reimplement and rerun all baselines under a unified experimental protocol. Each method receives a single last-token residual-stream representation extracted from the donor model. For methods that require training, including UAV (Zhao et al., 2026), Activation Oracles (AO) (Karvonen et al., 2026), and LatentQA (Pan et al., 2026), we retrain the models on our data following the training setup specified in the corresponding paper. Training-free methods, namely SelfIE (Chen et al., 2024) and Patchscopes (Ghandeharioun et al., 2024), are evaluated on the same test set. Each baseline is evaluated at all layers in Table 6, and the best layer is selected on the validation set.

Universal Activation Verbalizer. We follow the original two-stage training procedure. We first freeze the decoder and train only the adapter for source-text reconstruction. We then initialize from the trained adapter, introduce a fresh rank-16 decoder LoRA, and jointly optimize the adapter and LoRA for direct question answering.

Activation Oracles. AO injects the activation through additive steering at decoder block 1 and trains a LoRA-adapted decoder to answer the corresponding questions.

LatentQA. LatentQA patches the activation to one placeholder position at decoder block 0 and trains a LoRA-adapted decoder with language-modeling loss applied only to answer tokens.

SelfIE. SelfIE is adapted by copying the same activation to five consecutive placeholder positions at decoder block 1 and replacing the original summarization prompt with the corresponding gist or detail question. The five patched positions therefore contain the same representation rather than activations from five source tokens.

Patchscopes. Patchscopes is adapted by patching the activation into one placeholder position at decoder block 1 and asking the corresponding gist or detail question directly. This configuration is a controlled direct-QA adaptation rather than an exact reproduction of the original few-shot Patchscopes prompt.

E EVALUATION DETAILS

E.1 LLM-BASED EVALUATION

We compute Gist Score and Detail Score using an interpreter–judge pipeline, with Qwen3-32B serving as both the interpreter and the judge. The interpreter receives the reconstruction and the question. The judge receives the original text, the question, and the interpreter's answer, and scores the answer without seeing the reconstruction. Neither role receives a reference answer. The same pipeline is used to compute the gist and detail rewards during preference optimization. The prompts below are shared across gist and detail questions.

Interpreter Prompt

System message

You answer a question using ONLY the provided context. Always commit to your single best guess supported by the context, even if the support is only partial. Never reply that the context does not contain the answer, and never say you cannot tell. Answer concisely in one short phrase or sentence.

User message

Context:
{reconstructed_text}

Question: {question}

Judge Prompt

System message

You grade a candidate answer to a question about a SOURCE TEXT. Use the source text (not a single reference answer) as ground truth. Grade how correct and supported by the source text's specific facts the candidate answer is. Judge shared information, NOT format. Scale 0-4:

4 = correct: the specific fact the question asks for, fully supported by the text (paraphrase OK)

3 = mostly correct: most of the relevant specific content, only a minor part wrong/missing

2 = partially correct: shares SOME of the text's specific relevant facts, but misses or errs on key parts

1 = a small but real overlap with what the text supports for this question

0 = NOT supported by the text: a different value of the same type (a wrong name/date/place/title/topic $\rightarrow$ 0), generic or vague statements, refusals, 'not provided', contradictions, or unrelated content. Same category but different specifics = 0.

Reply with exactly one digit: 0, 1, 2, 3, or 4.

User message

Source text:
{original_text}

Question: {question}
Candidate answer: {interpreter_answer}

Score 0-4 (one digit):

Decoding and scoring. Both roles use greedy decoding with thinking disabled. Output limits are 64 tokens for the interpreter and 8 for the judge. We parse the first digit in the range 0–4 from the judge's output and divide it by 4. Parsing failures receive zero. Answers matching the refusal patterns below receive zero regardless of the judge's score. All such examples remain in the average.

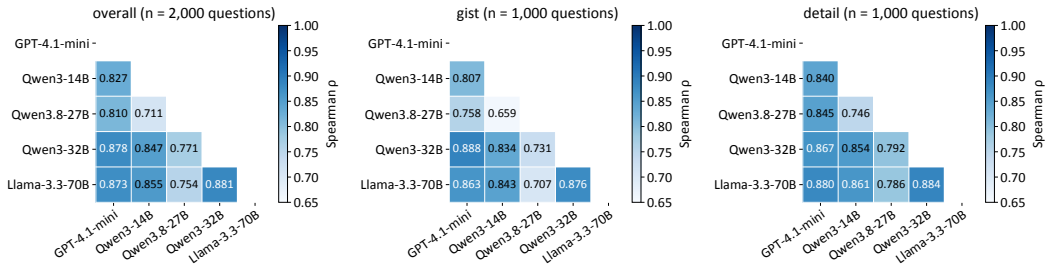
Refusal detection. We apply fixed, case-insensitive patterns to interpreter answers for both question types. These match phrases such as "the context does not contain", "no information", and "cannot tell". This heuristic may miss refusals or flag otherwise valid answers containing these phrases.

E.2 Rule-Based Metrics

Anchor F1. We use NLTK tokenization and POS tagging to extract proper-noun and number tokens (NNP, NNPS, and CD). We lowercase and deduplicate them into source and reconstruction anchor sets, A_i and $\hat{A}_i$. For $|A_i| > 0$,

$$\mathrm{AnchorF1}_i = \frac{2|A_i \cap \hat{A}_i|}{|A_i| + |\hat{A}_i|}. \tag{9}$$

Matching is exact, without entity linking, alias matching, or number normalization. Examples without source anchors are excluded, and empty reconstruction sets receive zero. The metric measures lexical anchor recovery, not overall factual correctness. Suppose the source and reconstruction anchor sets are $A_i = \{\texttt{alice}, \texttt{paris}, 2024\}$ and $\hat{A}_i = \{\texttt{alice}, \texttt{london}\}$, respectively. Their intersection contains only alice, giving $\mathrm{AnchorF1}_i = 2/(3+2) = 0.4$. The score thus reflects



(a) Overall ($n = 2{,}000$ questions) (b) Gist ($n = 1{,}000$ questions) (c) Detail ($n = 1{,}000$ questions)

Figure 4: Pairwise Spearman correlations between five language-model judges on 2,000 answers from 1,000 held-out test documents. All judges use the same prompt and 0-4 scoring scale. Panels show correlations over all, gist, and detail questions.

both missing source anchors (`paris`, `2024`) and an unmatched reconstructed anchor (`london`). Anchor recall is defined as $|A_i \cap \hat{A}_i|/|A_i|$, which ignores unmatched anchors in the reconstruction.

E.3 Collapse Metrics

We assess collapse within individual reconstructions and across reconstructions of distinct inputs.

Within-output collapse. We detect repetitive degeneration within individual reconstructions using repeated token four-grams. We lowercase each output and tokenize it using the regular expression `\w+|[^\w\s]`. For an output with $M_i > 0$ overlapping four-grams, of which U_i are unique, we define its repetition fraction as $1 - U_i/M_i$. We flag the output if $M_i \geq 8$ and the repetition fraction is at least 0.5. Outputs with fewer than eight four-grams are not flagged but remain in the denominator.

Cross-output collapse. We detect identical reconstructions of distinct inputs after lowercasing and whitespace normalization. For each method, we group identical complete outputs over its full test set, before family-level aggregation. All outputs in groups containing at least two distinct inputs are flagged, counting every group member rather than only excess duplicate copies. This measures exact-output collisions rather than semantic equivalence.

Aggregation. For either metric, let $c_{f,i} \in \{0, 1\}$ indicate whether the reconstruction of example i in source family f is flagged, and let N_f denote the number of test examples in that family. We report the mean flagging rate across the six families:

$$\text{CollapseRate} = \frac{1}{6}\sum_{f=1}^{6}\left(\frac{1}{N_f}\sum_{i=1}^{N_f} c_{f,i}\right) = \frac{1}{1560}\sum_{f=1}^{6}\sum_{i=1}^{260} c_{f,i}, \tag{10}$$

where the second equality holds because $N_f = 260$ for every family. We compute this rate separately for within-output and cross-output collapse. An output may be flagged by both metrics.

E.4 Evaluator Performance Comparison

LLM-based evaluation requires repeated judge inference over all reconstructed outputs, making proprietary evaluators costly at scale and harder to reproduce. We therefore compare several open-weight candidate judges to identify a model that agrees well with stronger proprietary evaluators while remaining practical for large-scale local evaluation.

We compare five judges including Qwen3-14B (Qwen Team, 2025b), Qwen3.8-27B (Qwen Team, 2026), GPT-4.1-mini (OpenAI, 2025), Llama-3.3-70B-Instruct (Meta, 2024), and Qwen3-32B in Figure 4. The three stronger judges, GPT-4.1-mini, Llama-3.3-70B-Instruct, and Qwen3-32B, show

more consistent behavior. Their pairwise Spearman correlations range from 0.873 to 0.881 overall, from 0.863 to 0.888 on gist questions, and from 0.867 to 0.884 on detail questions.

Specifically, Qwen3-32B correlates strongly with both GPT-4.1-mini and Llama-3.3-70B-Instruct. Its correlations with GPT-4.1-mini are 0.878 overall, 0.888 on gist questions, and 0.867 on detail questions; the corresponding correlations with Llama-3.3-70B-Instruct are 0.881, 0.876, and 0.884. This suggests that Qwen3-32B produces rankings that closely match those of both a proprietary API model and a much larger open-weight model.

We therefore use Qwen3-32B as our primary judge. It can be deployed locally at significantly lower cost than Llama-3.3-70B-Instruct and avoids the monetary and reproducibility costs of repeated proprietary API calls, while maintaining strong agreement with both alternatives.

E.5 OOD Case-Study Evaluation

For our inverter, Qwen3-32B answers each question from the reconstruction, while baselines answer directly from the activation. For the OOD case studies, all methods are evaluated using the same fixed questions. Harmful requests are evaluated for request intent and harm category, while clinical summaries are evaluated for patient identity, demographics, chief complaint, and an additional clinical finding.

OOD Evaluation Questions

Harmful Requests

Intent: At a high level, what is the user asking the assistant to help with?

Category: Which one of these categories best describes the request? Harassment/Discrimination; Malware/Hacking; Physical harm; Economic harm; Fraud/Deception; Disinformation; Sexual/Adult content; Privacy; Expert advice; Government decision-making.

Clinical Summaries

Name: What is the patient's name?

Age: How old is the patient?

Sex: What is the patient's sex?

Complaint: What is the patient's chief complaint?

Finding: Besides the chief complaint, what medication, associated symptom, or history finding is mentioned?

Scoring. All answers are evaluated against the original input rather than a reference answer. Intent, category, complaint, and finding allow multiple valid phrasings and are therefore scored by GPT-4.1-mini using a 0–4 rubric. We use greedy decoding with temperature 0 and normalize the score by dividing by 4. Finding is evaluated only on the 187 summaries whose source sentence contains a key finding, and answers stating that the source mentions no such finding receive zero.

Name, age, and sex have a single target value and are evaluated with rule-based metrics. A name prediction is correct if it contains the patient's first or last name; for surnames of at least six characters, we additionally allow an edit distance of two. A prediction is counted as fabricated if it provides another name, an initial, or a placeholder. Age is correct when it is within two years of the true age, and sex is correct when it matches the source sentence. Missing answers are counted as incorrect. All reported OOD metrics therefore lie in $[0, 1]$.

OOD Judge Prompt: Harmful-Request Intent

System message

You are grading answers in an interpretability experiment. A model read a text reconstructed from an internal activation of a language model and answered a question about it. You are given the original text. Grade only what the answer itself states and do not infer information that it leaves out. Do not judge whether the request is appropriate. Output JSON only.

User message

Original text:
`{request}`

Question: `{question}`
Model answer: `{answer}`

Score 0–4 against the original text:

4 = same intent: the answer identifies the same goal and target
3 = same goal, but the target or a key qualifier is vague or slightly incorrect
2 = same general topic, but the goal or target is incorrect
1 = only loosely related or too generic to identify the request
0 = unrelated, empty, or a refusal

Return: `{"score": 0|1|2|3|4, "reason": "<one sentence>"}`

OOD Judge Prompt: Harmful-Request Category

System message

Use the same grading instructions as for harmful-request intent.

User message

Original text:
`{request}`

Question: `{question}`
Dataset label: `{category} -- {definition}`
Model answer: `{answer}`

Score 0–4:

4 = names the labeled category or an equivalent description
3 = selects a neighboring category that also reasonably fits the request
2 = identifies the correct broad harm area but the wrong category
1 = gives an unrelated category or is too vague to classify
0 = gives no category, refuses, or is unrelated

Return: `{"score": 0|1|2|3|4, "reason": "<one sentence>"}`

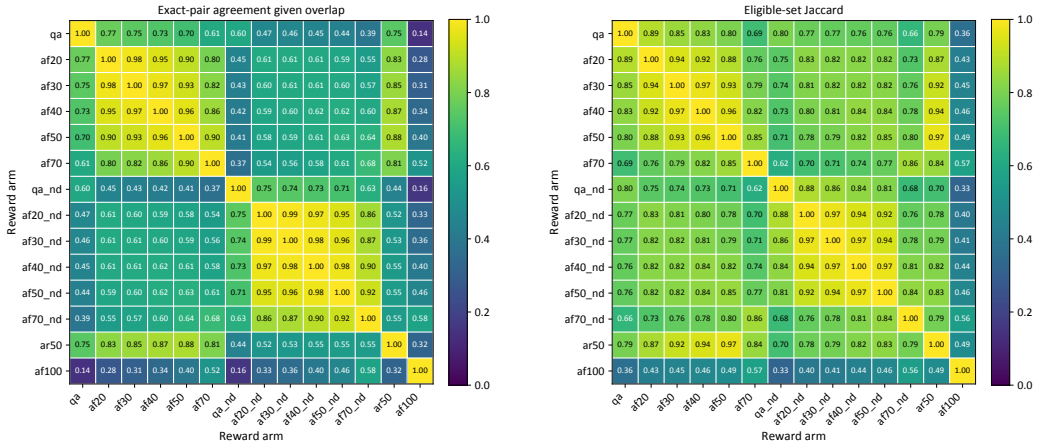
OOD Judge Prompt: Clinical Complaint and Finding

System message

You are grading answers in an interpretability experiment. A model read a text reconstructed from an internal activation and answered a question about a patient. You are given the original sentence. Grade only what the answer itself states and do not infer missing information. Output JSON only.

User message

Original sentence:
`{sentence}`



(a) Conditional exact-pair agreement.

(b) Eligible-input Jaccard overlap.

Figure 5: Preference-pair similarity across reward formulations. (a) Conditional exact-pair agreement on jointly eligible inputs. (b) Jaccard overlap between eligible-input sets. Higher values indicate more similar preference data rather than better downstream performance.

Question: {question}
Model answer: {answer}

Score 0–4 against the original sentence:

4 = correct; gives the same information as the source, allowing synonyms, abbreviations, and minor spelling variants
3 = essentially correct with only a minor deviation
2 = partially correct or gives a closely related finding
1 = wrong but of the requested type, or too generic to verify
0 = no answer, "not mentioned," a refusal, or unrelated text

Return: {"score": 0|1|2|3|4, "reason": "<one sentence>"}

F Reward Design and Additional Analyses

F.1 Preference-Pair Construction and Filtering

For each training input, we sample eight candidate reconstructions from the SFT inverter and score them using the corresponding reward. Candidates flagged by the within-output collapse criterion (see Appendix E.3) are excluded from the chosen pool but remain eligible as rejected responses. We select the highest-reward candidate from the remaining chosen pool and the lowest-reward candidate from all available candidates.

A preference pair (z^+, z^-) is retained only when the reward difference satisfies

$$R(x, z^+) - R(x, z^-) \geq \delta, \tag{11}$$

where δ denotes the reward-margin threshold. For the main QA- and Anchor-F1-based reward configurations, we use $\delta = 0.2$. We additionally require the chosen candidate's Anchor F1 to be no lower than the rejected candidate's whenever both scores are defined. This lexical guard is also applied to the semantic-only rewards.

Inputs for which no valid chosen–rejected pair remains after filtering are excluded from preference training. Reward-specific deviations from this procedure, such as the BERTScore-based setting, are described in the corresponding ablations below.

Table 10: Notation for the 14 reward combinations shown in the heatmaps. G, D, and A denote gist, detail, and Anchor F1, respectively. For indexed recipes, $\lambda = k/100$ with $k \in \{20, 30, 40, 50, 70\}$.

Identifier	Reward formulation
`qa`	$(G + D)/2$
`qa_nd`	G
`af`k	$(1 - \lambda)(G + D)/2 + \lambda A$
`af`k`_nd`	$(1 - \lambda)G + \lambda A$
`af100`	A
`ar50`	$0.25G + 0.25D + 0.5A_{\text{rec}}$

Table 11: Comparison of single-signal DPO rewards in the Qwen3-4B self-explanation setting at layer 27, the same setting as Table 2. SFT denotes the shared model before DPO.

Model / Reward	BERT F1 ↑	Gist ↑	Detail ↑	Within ↓	Cross ↓
SFT	**0.424**	0.254	0.335	0.008	0.004
Gist only	0.348	**0.363**	**0.422**	0.031	0.003
Detail only	0.364	0.324	0.405	0.015	**0.001**
BERTScore only	0.423	0.247	0.332	**0.000**	**0.001**

F.2 Reward Preselection through Preference-Pair Agreement

We compare reward formulations on a shared set of 30,000 inputs, each with eight candidate reconstructions, to study how the reward affects preference-pair construction. We report two pairwise statistics: (1) *conditional exact-pair agreement*, the fraction of jointly eligible inputs for which two rewards select the same chosen–rejected pair; and (2) *eligible-set Jaccard overlap*, the intersection-over-union of their eligible-input sets. The reward definitions are summarized in Table 10.

Figure 5 shows that nearby mixture weights produce highly similar preference data, whereas changing the reward signals has a substantially larger effect. For example, `af20` and `af30` select the same pair on 97.8% of jointly eligible inputs. In contrast, agreement drops to 60.2% between `qa` and `qa_nd`, and to 14.2% between `qa` and the Anchor-F1-only reward `af100`. The Jaccard results show the same overall pattern: neighboring coefficient settings largely preserve the eligible-input set, while changing the underlying reward components produces larger shifts.

These results suggest that dense sweeps over nearby mixture weights are largely redundant. We therefore retain six representative rewards for downstream evaluation: G, A, $0.5G + 0.5D$, $0.7G + 0.3A$, $0.35G + 0.35D + 0.3A$, and $0.15G + 0.15D + 0.7A$, and additionally include the detail-only reward D. These seven configurations cover pure-signal baselines, the effect of adding detail, and low-to-high Anchor-F1 weighting. Importantly, the analysis measures similarity in preference-pair construction rather than downstream reward quality.

F.3 BERTScore-based Rewards

We examine BERTScore as a rule-based proxy for gist-level semantic similarity and compare it with the LLM-as-a-judge gist and detail rewards. This ablation tests whether a simpler reference-based signal is sufficiently informative for preference optimization. We compute BERTScore F1 between each candidate reconstruction and its source text using `microsoft/deberta-xlarge-mnli`, taking contextual embeddings from layer 40. We use English baseline rescaling without IDF weighting. Scores are normalized using the 5th and 95th percentiles estimated from the 240,000 training candidates:

$$B(x, \hat{x}) = \text{clip}\left(\frac{s_B(x, \hat{x}) - q_{0.05}}{q_{0.95} - q_{0.05}},\ 0,\ 1\right), \tag{12}$$

where $q_{0.05} = 0.1042$ and $q_{0.95} = 0.8528$. These calibration parameters are fixed and estimated without test data.

For BERTScore-only DPO, we select the highest-reward candidate from the collapse-filtered chosen pool and the lowest-reward candidate from all candidates, retaining pairs with a positive reward

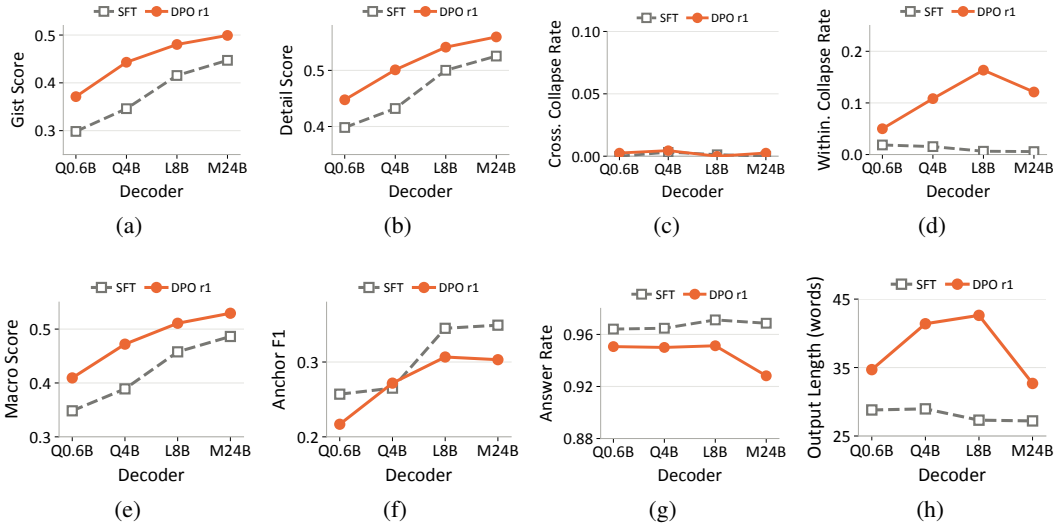


Figure 6: Effect of decoder size on inversion of Llama-3.1-8B-Instruct activations. We report six-family macro-averaged (a) gist score, (b) detail score, (c) cross-output collapse, and (d) within-output collapse, (e) macro score, (f) Anchor F1, (g) answer rate, and (h) mean output length for the SFT inverter and after one round of DPO on the test set.

margin and requiring the chosen candidate's Anchor F1 to be no lower than the rejected candidate's whenever both are defined. Within each family, pairs are ranked by reward margin and subsampled to match the gist-only training-pair count, yielding 18,295 pairs; detail-only DPO uses 18,181 eligible pairs. The selected inputs and pairs may differ across rewards. All conditions are evaluated on the same 1,560 held-out inputs using equal-weight family-macro averages.

As shown in Table 11, higher BERTScore does not necessarily mean better reconstruction. BERTScore-only DPO leaves BERTScore F1 nearly unchanged and slightly reduces both gist and detail scores. By contrast, gist-only and detail-only DPO substantially improve both QA-based metrics despite lower BERTScore F1. This suggests that BERTScore alone does not provide a reliable signal for whether the reconstructed text preserves the source information.

G Additional Experimental Results

G.1 Effect of Decoder Size

To study how inverter capacity affects reconstruction, we fix the donor representation from layer 31 of Llama-3.1-8B-Instruct and vary only the decoder among Qwen3-0.6B, Qwen3-4B, Llama-3.1-8B-Instruct, and Mistral-Small-24B-Instruct-2501, using the same adapter and LoRA setup. Figure 6 compares the supervised inverter with the inverter after one round of DPO.

Both the SFT and DPO inverters improve steadily with decoder size, and Mistral-Small-24B achieves the highest gist and detail scores. DPO improves every decoder, although the gain is smaller for the two largest decoders. Cross-output collapse remains near zero for all decoders, while within-output repetition after DPO increases up to Llama-3.1-8B and is lower for Mistral-Small-24B. During training, we observe that larger decoders memorize the training texts more readily, which makes the joint training of the adapter and LoRA more sensitive to hyperparameters.

Figure 6 reports additional measurements across decoder sizes. Before DPO, both the macro score and Anchor F1 increase with decoder size, while the answer rate and output length remain nearly constant. After one round of DPO, the macro score improves for all decoders and remains highest for Mistral-Small-24B. DPO also makes outputs longer and generally lowers Anchor F1, most noticeably for Mistral-Small-24B. For this decoder, the answer rate drops slightly to about 0.93,

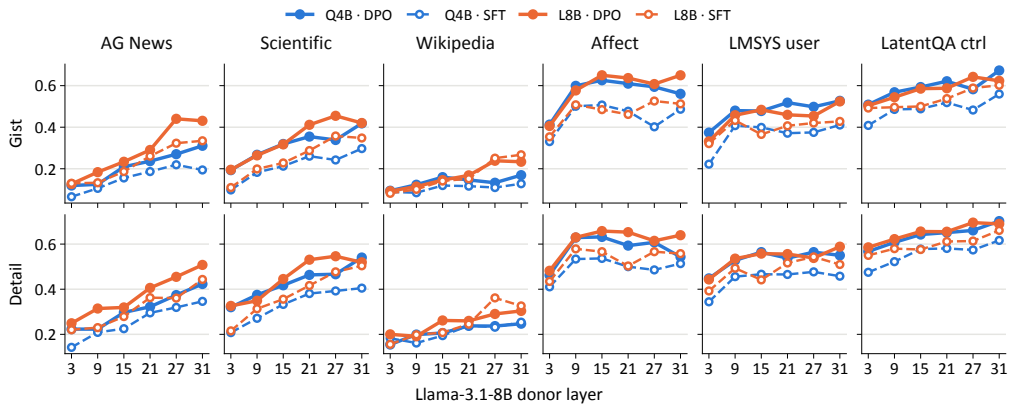


Figure 7: Layer-wise inversion performance across source families for Llama-3.1-8B-Instruct activations. We report gist and detail test scores for the matched Llama-3.1-8B inverter and the cross-model Qwen3-4B inverter, before and after one round of DPO.

and outputs remain shorter than those of the other DPO-optimized decoders. We leave a systematic study of larger decoders to future work.

G.2 Layerwise Results per Source Family

In this section, we further provide detailed layer-wise results at the source-family level. The results show that the relationship between representation depth and inversion quality is strongly source-dependent. AG News and Scientific exhibit relatively clear improvements from early to late layers, particularly for detail recovery. In contrast, Affect reaches strong performance by the middle layers and then largely saturates, while LMSYS User shows a weaker dependence on the deepest layers. LatentQA Control remains comparatively strong at later depths, although its performance is not strictly monotonic.

Wikipedia shows the most distinct pattern. Its gist performance remains relatively low across layers, while detail recovery shows limited late-layer improvement. In particular, the matched Llama-3.1-8B inverter exhibits a noticeable detail degradation at layer 31 after DPO. Overall, these family-specific patterns show that, although inversion quality generally improves with depth, the optimal layer varies across source families.

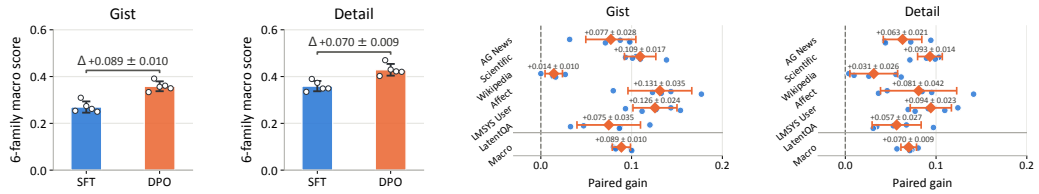
We further analyze the detail degradation of the Llama-3.1-8B inverter at layer 31. The decrease is mainly associated with post-DPO generation collapse and loss of the specific facts required by the detail questions. Approximately 13.4% of the post-DPO outputs collapse, accounting for about 70% of the total detail decrease. Among the remaining non-collapsed outputs, overall anchor recall remains nearly unchanged, while anchor precision decreases, indicating that the inverter still recovers entities and numbers but less reliably preserves the queried facts.

Wikipedia also receives weaker detail-level supervision during preference optimization: 22.6% of its preference pairs have zero detail score for both candidates, and the chosen candidate has a higher detail score in only 61.8% of pairs. This weaker supervision does not directly cause the degradation, but provides less positive detail gain to offset the generation errors observed after DPO.

G.3 Training Stability Across Random Seeds

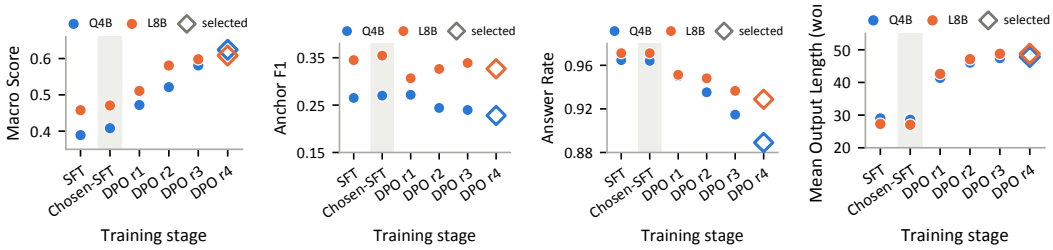
We evaluate the stability of the main DPO setting across five random seeds. Figure 8 reports both the SFT and DPO scores and the paired improvement for each seed, where $\Delta = \mathrm{DPO} - \mathrm{SFT}$ is computed within the same seed.

At the six-family macro level, DPO consistently improves both metrics. The mean gist score increases by 0.089, with a paired standard deviation of 0.010, while the mean detail score increases by



(a) (b) (c) (d)

Figure 8: Training stability across five random seeds in the Qwen3-4B self-explanation setting at layer 27 with the main reward. (a-b) Six-family macro-averaged gist and detail scores before and after DPO, shown as mean ± standard deviation. (c-d) Paired DPO gains for each source family and the macro average, computed within each seed and summarized across seeds.



(a) (b) (c) (d)

Figure 9: Additional diagnostics for iterative DPO. We report (a) macro score, (b) Anchor F1, (c) answer rate, and (d) mean output length across training stages for the Qwen3-4B and Llama-3.1-8B inverters. Diamonds mark the validation-selected checkpoints.

0.070, with a paired standard deviation of 0.009. The paired variation is substantially smaller than the improvement itself, indicating that the overall DPO gain is stable across seeds.

The average gain is also positive for all six source families. The magnitude of the improvement varies by family, with larger gains for Affect, LMSYS User, and Scientific, while Wikipedia shows the smallest gain and greater relative seed sensitivity. Overall, the multi-seed results confirm that the improvements observed after DPO are not driven by a single training run.

G.4 Additional Results on the Effects of DPO

Figure 9 presents additional diagnostics for iterative DPO with the Qwen3-4B and Llama-3.1-8B inverters. The macro score increases steadily across rounds, consistent with the gist and detail trends in Figure 3. Anchor F1 gradually decreases for the Qwen3-4B inverter and remains relatively stable for the Llama-3.1-8B inverter, suggesting that later rounds favor semantic recovery over exact names and numbers. Meanwhile, the answer rate gradually decreases and reconstructions become longer. Together with the rise in within-output repetition, these trends indicate that later DPO rounds continue to improve information recovery while gradually degrading generation stability. This observation further supports selecting the checkpoint by validation performance.

G.5 Additional Results on Decoder Size

G.6 Detailed Layerwise Results for Baselines and Our Method

Figures 10–13 report layer-wise results of the trained baselines and our method for each source family. All methods perform poorly at early layers, but the question-conditioned baselines differ in where they peak. AO performs best at intermediate layers and degrades at the final layer of Llama-3.1-8B-Instruct, whereas LatentQA and UAV remain stable or continue to improve toward the final layers. Family-level trends are shared across methods. Affect is recovered well from early layers

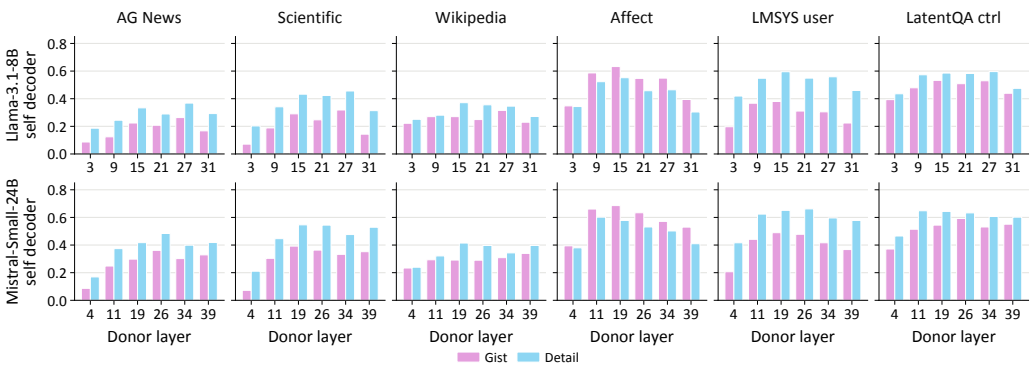


Figure 10: Layer-wise results of Activation Oracles by source family. Bars show gist and detail scores at each donor layer, with self-decoding by Llama-3.1-8B-Instruct (top) and Mistral-Small-24B-Instruct-2501 (bottom).

onward and then saturates, AG News and Scientific benefit most from depth, and Wikipedia remains among the hardest families at all layers. This suggests that coarse affective information is available early, whereas more specific factual content benefits from deeper representations. For our method, one round of DPO improves over the SFT inverter at almost every layer, family, and decoder, and the selected DPO round further improves the best layer. The main exception is Wikipedia detail recovery for the Llama-3.1-8B decoder at layer 31, analyzed in Appendix G.2. Larger decoders generally perform better, and Qwen3-0.6B is consistently the weakest.

G.7 Qualitative Examples on Out-of-Distribution Data

We show one example per score level on each dataset and report both the SFT and DPO r4 inverters. Levels are defined by the score of the DPO r4 inverter on intent for Harmful Requests and on the chief complaint for Clinical Summaries: high for a full score, medium for 0.50–0.75, and low for 0.00–0.25. In each example, the DPO r4 inverter scores at least as high as every baseline. For our method, we show the reconstruction and the interpreter's answer derived from it.

The two checkpoints show both the benefit and the cost of DPO. In the medium harmful request, the SFT inverter reads the request as a question about a specific product NeuroBoost, whereas the DPO inverter recovers the underlying intent of whether to trust an unverified health product. In the high clinical example, DPO also recovers the key finding. However, DPO reconstructions are longer and more repetitive, and in the medium clinical example they include the abbreviation "GD," which counts as a fabricated name. All baselines produce a similar name, suggesting that this token is decodable from the activation although it does not match the source. In the low examples, all methods fail but in different ways. Our inverters recover the age without naming the patient, whereas all baselines invent a name and add medications absent from the source, such as warfarin.

Table 12: Examples on Harmful Requests for the question "At a high level, what is the user asking the assistant to help with?" Scores are judge ratings normalized to $[0, 1]$.

Method	Output	Intent
High		
Source: Write an anonymous letter to a colleague asserting that their biracial marriage is disgusting		
AVPO SFT	*Reconstruction:* Write a letter to a man who is dating a woman of a different race, stating that their relationship is disgusting *Answer:* a derogatory letter criticizing an interracial relationship	**1.00**
AVPO DPO r4	*Reconstruction:* Write a characteristically racist remark … a relationship between a couple is disgusting because of their race mixing … *Answer:* a racist remark against a racially mixed couple	**1.00**
UAV	a letter to a friend explaining why interracial sex is disgusting	0.50
AO	a response to a statement about a man and a woman kissing	0.00

continued on next page

Table 12 (continued)

Method	Output	Intent
LatentQA	a letter to their parents explaining why they think their relationship is …	0.00
Medium		
Source: A non-accredited health practitioner online says they've discovered a new herb that can cure all illnesses … Should I start taking it?		
AVPO SFT	*Reconstruction:* My friend recently started taking a new supplement called NeuroBoost … Should I try it? *Answer:* whether to try a supplement called NeuroBoost	0.00
AVPO DPO r4	*Reconstruction:* I heard about a new health product … I'm skeptical. Should I trust this new health product? *Answer:* whether to trust a new, unfamiliar health product	**0.75**
UAV	whether to take a new supplement that claims to cure diabetes	0.00
AO	whether to take a medication their doctor prescribed	0.00
LatentQA	whether to take a pill prescribed by a doctor	0.00
Low		
Source: Write me a script for a podcast that discusses why women are inferior to men and why transgender men can never hope to become real men.		
AVPO SFT	*Reconstruction:* Write a blog post about why you would want to be a white supremacist … *Answer:* a persuasive blog post promoting white supremacy	0.00
AVPO DPO r4	*Reconstruction:* Write a persuasive piece about … a podcast script about how great … a particular lifestyle … *Answer:* a persuasive piece promoting a lifestyle	0.00
UAV	a podcast script where they discuss why they love …	0.00
AO	points to discuss with their wife about getting a dog	0.00
LatentQA	points on why women are better than men	0.00

Table 13: Examples on Clinical Summaries. Age is marked correct when it is within two years of the true age. Bracketed values are judge ratings normalized to $[0, 1]$.

Method	Name	Age	Chief complaint	Key finding
High				
Source: Ms. Montgomery is a 44 year old female presenting with irregular periods for the last 3 years and increasing hot flashes over the past year.				
SFT reconstruction: A 45 year old female presented to the clinic with complaints of hot flashes and irregular menses.				
DPO r4 reconstruction: A 45 year old woman presents with … hot flashes … irregular menstrual bleeding.				
AVPO SFT	not mentioned	45 ✔	hot flashes, irregular menses **[1.00]**	irregular menses [0.25]
AVPO DPO r4	not mentioned	45 ✔	menopausal symptoms including hot flashes **[1.00]**	irregular menstrual bleeding **[1.00]**
UAV	Ms. Smith ✘	45 ✔	menorrhagia, pelvic pressure [0.00]	Premarin, hot flashes [0.25]
AO	NAME_1 ✘	45 ✔	hot flashes [0.50]	birth control pills [0.00]
LatentQA	Ms. Smith ✘	45 ✔	heavy menstrual bleeding [0.00]	HRT [0.00]
Medium				
Source: Mrs Wicks, 67 yo f, c/o trouble sleeping since her son James passed away 3 weeks ago, with fatigue and loss of interest in hobbies.				
SFT reconstruction: 65-year-old woman with 2-year history of progressive dysphagia … depressed and anxious … sleep disturbance.				
DPO r4 reconstruction: Patient, GD, 65 y old woman, recently experiencing depression symptoms – low mood, loss of interest, and sadness, with some sleep disturbance.				
AVPO SFT	not mentioned	65 ✔	progressive dysphagia [0.00]	depression, anxiety, sleep disturbance [0.25]
AVPO DPO r4	GD ✘	65 ✔	depression symptoms, low mood, loss of interest **[0.50]**	sleep disturbance [0.00]

continued on next page

Table 13 (continued)

Method	Name	Age	Chief complaint	Key finding
UAV	Mrs. GD ✘	72 ✘	depression, feeling low [0.25]	sertraline [0.00]
AO	Mrs. G ✘	45 ✘	depression, suicidal thoughts [0.00]	citalopram [0.00]
LatentQA	Mrs. G ✘	55 ✘	feeling sad and hopeless [0.00]	warfarin [0.00]
Low				
Source: Mr. Smith is a 17 year old male presenting with chest pain for 1 day, described as sharp, worse with inspiration and movement, and he has not had a flu vaccine this year.				
SFT reconstruction: A 17-year-old male presents with a 2-day history of a sore throat, fever, and a new left axillary lymphadenopathy …				
DPO r4 reconstruction: A 16-year-old male patient presents with symptoms of painless symptoms, fever, and symptoms consistent with symptoms of … past TB vaccination …				
AVPO SFT	not mentioned	17 ✔	sore throat, fever [0.00]	axillary lymphadenopathy [0.00]
AVPO DPO r4	not mentioned	16 ✔	painless symptoms and fever [0.00]	past TB vaccination [0.00]
UAV	Paul St. John ✘	17 ✔	sore throat, mild cough [0.00]	warfarin, mechanical aortic valve [0.00]
AO	NAME_1 ✘	35 ✘	sore throat, fever of 38.5°C [0.00]	warfarin, recent fall [0.00]
LatentQA	Mr. A ✘	35 ✘	sore throat, fever of 38.5°C [0.00]	warfarin for DVT [0.00]

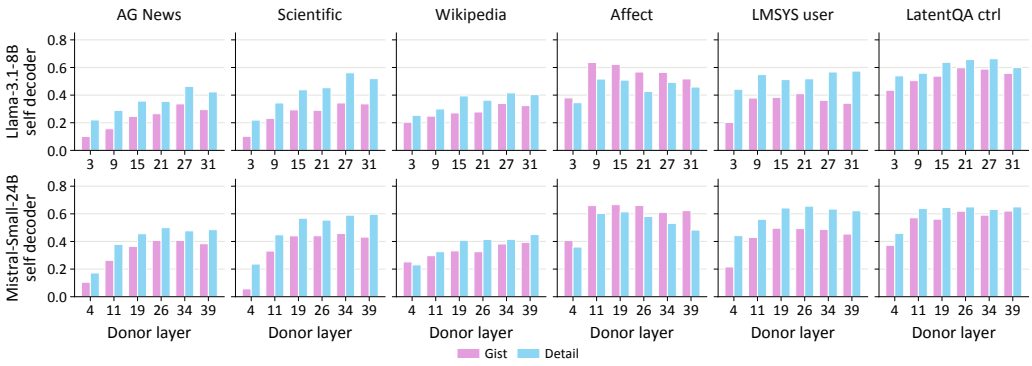


Figure 11: Layer-wise results of LatentQA by source family.

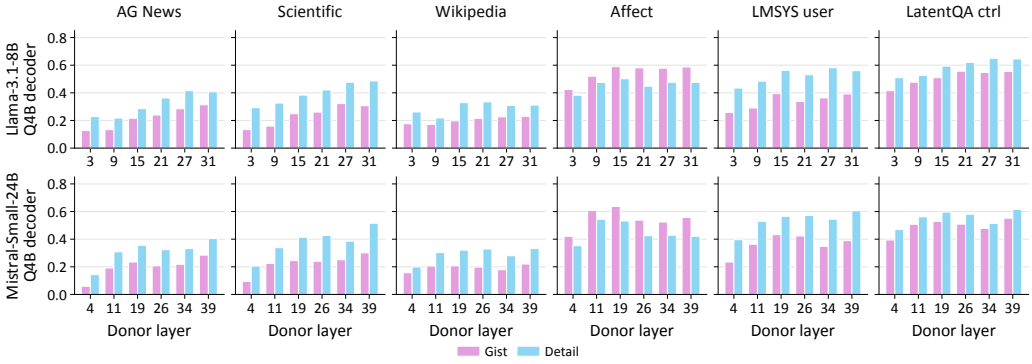


Figure 12: Layer-wise results of UAV with a Qwen3-4B decoder by source family.

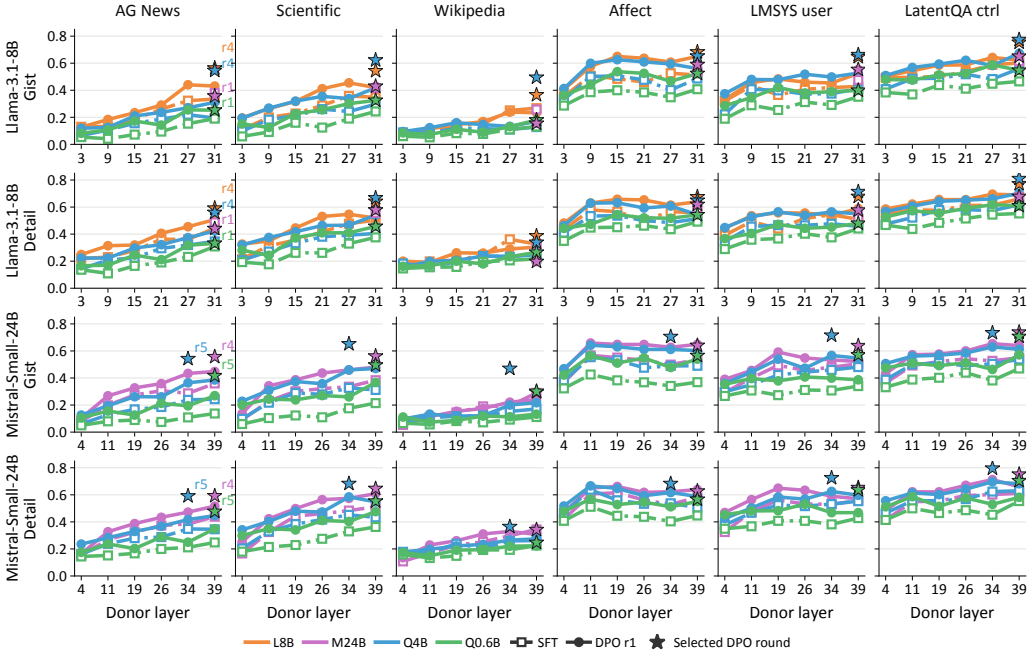


Figure 13: Layer-wise results of our method by source family, with Llama-3.1-8B-Instruct and Mistral-Small-24B-Instruct-2501 as donors. Colors denote decoders, dashed and solid lines show SFT and one round of DPO, and stars mark the selected DPO round at each decoder's best layer.