SELF-SUPERVISED PREFERENCE LEARNING FOR MULTIMODAL FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Preference optimization for multimodal foundation models typically relies on external judgment, such as human annotations, model-based scoring, or task-specific supervision. We introduce structural-neighbor preference learning, a self-supervised method based on a simple principle: structurally similar inputs should receive similar descriptions. For each image in a data set of interest, we identify others with similar underlying patterns–for example, charts with similar trends and fluctuations–and collect the model's descriptions of the image and its neighbors. We rank these candidate descriptions by how much they agree with descriptions of similar images and disagree with those of dissimilar images, selecting the highest- and lowest-ranked candidates as preferred and rejected responses. We apply this method to DPO on Qwen2-VL-7B and Gemma-3-4B, testing across financial price paths, electrocardiograms, and seismic traces. We find the preference signals our approach learns transfer zero-shot to held-out tasks: relative to the supervised finetuned baseline, the detection of large price moves increases by 31%, crash-risk prediction by 14%, electrocardiogram abnormality detection by 10%, and seismic event detection by 3.3%. Our method also outperforms image conditioned, self-rewarding, and external-judge preference baselines, and remains effective when similarity between inputs is determined from image-derived representations rather than underlying signal. We also observe improvements on other types of images, including histograms, scatter plots, and natural images.

1 INTRODUCTION

Preference optimization has become a standard tool for improving multimodal foundation models. Existing approaches typically construct preference pairs from external judgment sources, including human annotators, model-based scoring, or task-specific supervision. We ask whether useful preference supervision can instead come from relationships among unlabeled inputs themselves, stating that if two inputs are structurally similar, a well-functioning model should describe them similarly.

Most existing preference-learning methods generally score responses at the level of individual image-response examples, rather than asking whether the model behaves consistently across related images. As a result, models can learn to prefer responses that sound fluent without sufficiently relying on visual input, an instance of the unconditional preference problem (Wang et al., 2024a). Relationships across examples provide an alternative source of supervision: if two images are structurally similar, a visually grounded model should describe them similarly. This encourages preferences to reflect the visual input rather than linguistic priors alone.

We introduce *structural-neighbor preference learning*, a self-supervised method for constructing preference pairs from these cross-example relationships. For each image, we identify a set of similar inputs and, using the model, generate candidate descriptions for the image and its neighbors. The neighborhood can be defined using a representation that captures meaningful input structure; in our primary time-series experiments, we use the underlying signal, and later experiments demonstrate that image-derived representations also provide an effective signal. We rank these candidates by how well they agree with structurally similar inputs while remaining distinct from dissimilar ones: the highest-ranked candidate becomes the preferred output, the lowest-ranked the rejected output. We instantiate and test the method with direct preference optimization (DPO).

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

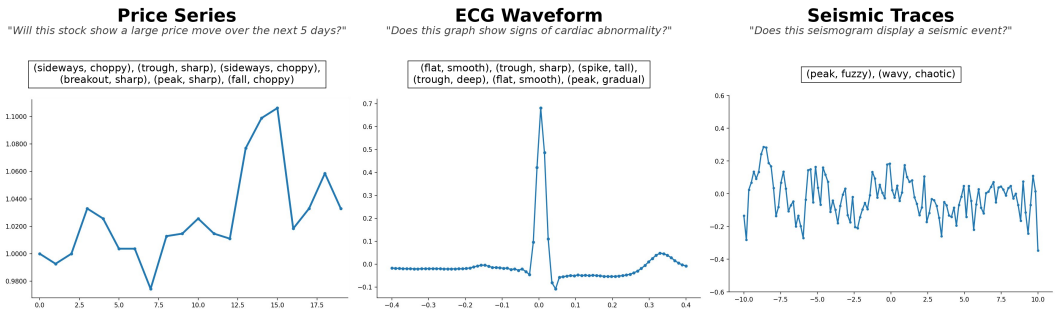


Figure 1: Three zero-shot transfer domains. Given only a time-series chart and a natural-language query, the model must answer a domain-specific question without receiving labels.

We test our method on two vision-language models, Qwen2-VL-7B and Gemma-3-4B, first across three unrelated time-series domains rendered as charts: financial price paths, electrocardiograms, and seismic traces. Time series provide a clean evaluation setting as they have defined downstream tasks and concentrate information in fine-grained visual structure—local extrema, trends, slope changes, oscillations—that fluent descriptions can easily miss. After three rounds of iterative DPO, we find the signal transfers zero-shot to held-out tasks that are tied to real-world phenomena, despite having never been trained on them. Relative gains over a supervised fine-tuning baseline reach 31% on large price-move detection, 14% on crash-risk prediction, 10% on electrocardiogram abnormality detection, and 3.3% on seismic event detection, with both models showing the same pattern. Probes show the gains reflect stronger alignment between visual inputs and textual outputs. Additional experiments show that the signal remains effective when neighborhoods are constructed from image-derived representations rather than raw signals, and it transfers beyond time-series charts to histograms, scatterplots, and photographs of birds.

We make three contributions:

1. We introduce structural-neighbor preference learning, a self-supervised method for constructing preference pairs from relationships among unlabeled inputs. It requires no human annotations, no external judges, and no task-specific supervision.
2. We show that these preferences improve zero-shot downstream performance across three time-series domains and two model families, and outperform alternative preference-learning baselines on the tasks evaluated.
3. We provide evidence that structural-neighbor preference learning improves visual grounding, and characterize how its effectiveness depends on the representation used to describe structural similarity. We also show that this method extends to a variety of domains.

2 Structural-Neighbor Preference Learning

We introduce a self-supervised procedure for constructing preference pairs from a collection of unlabeled images. It contains three steps: first, we generate a description of every image. Second, we identify images with similar underlying patterns, which we call structural neighbors. Third, we treat descriptions of each image and its neighbors as candidate responses for that image, then rank them to select preferred and rejected responses. We describe these steps below for time-series charts, using the examples in Figure 1.

2.1 Generating Chart Descriptions

We begin by prompting the model to describe patterns visible in each chart. For time series inputs, the model reads the chart from left to right as an ordered sequence of `(state, texture)` tuples, one per segment, which is a contiguous region of the chart, like a spike. The state captures the segment's overall shape, while the texture describes finer properties of that shape; for example, (sideways, choppy) describes an irregularly fluctuating segment with no clear upward or downward

trend, while (peak, sharp) describes a sharp peak. We use domain-relevant state and texture terms for each experimental setting, while keeping the representation and scoring procedure fixed across domains. This common format provides a simple and interpretable space in which candidate descriptions can be compared using edit distance. Figure 1 shows examples; prompts are in Appendix A.2. In our ablations, we evaluate alternative output formats (like free text) and different levels of prompt specificity.

2.2 Building the Structural Neighbor Graph

For each input x_i we identify a set of structurally similar neighbors. In our primary time-series experiments, we construct the neighborhood graph from the underlying signal s_i associated with each image. We mean-center and L_2 normalize each s_i, so that similarity reflects shape rather than absolute level or scale. We then compute cosine similarity between all the pairs of signals. Section 4 shows that neighborhoods can be alternatively constructed from image-derived representations when the underlying signal is unavailable. For each input, the structural neighbors, denoted as $N(i)$, are the top-K closest examples under this measure. We ensure that self-neighbors are removed by zeroing the diagonal of the similarity matrix before selecting neighbors. We exclude inputs whose neighborhoods are too weak to provide reliable supervision. If fewer than Y of the top-K neighbors exceed a fixed similarity threshold, we remove that input data point from the training set.

Similarly, we also sample a set of non-neighbors $R(i)$, which are structurally unrelated examples that define what the input description should remain distinguishable from. We obtain these with the same method, but from a set excluding the input image and all of $N(i)$. These non-neighbors are not random negatives: rather, they define what the anchor's description should remain distinguishable from. We use $K = 10$ for the size of $N(i)$ and $R(i)$ per input image.

2.3 Preference-based Scoring Function

For each input x_i, we generate a structured description d_i using greedy decoding at the start of each DPO round using the current policy. The candidate pool $\mathcal{C}_i$ consists of d_i together with the descriptions of its structural neighbors $\{d_j : j \in \mathcal{N}(i)\}$. We then rank candidates by scoring each candidate $c \in \mathcal{C}_i$ by how well it respects the local structure of the chart x_i: a good description should agree with neighbors and remain distinct from non-neighbors.

$$S(c \mid x_i) = -\frac{1}{Z_i} \sum_{j \in \mathcal{N}(i)} w_{ij} \, \mathrm{NED}(c, d_j) + \lambda \cdot \frac{1}{|\mathcal{R}(i)|} \sum_{k \in \mathcal{R}(i)} \mathrm{NED}(c, d_k)$$

The first term rewards agreement with structural neighbors. $\mathrm{NED}(c, d_j)$ is the normalized edit distance between the candidate and each neighbor description, weighted by w_{ij}, the cosine similarity between s_i and s_j, so that closer neighbors exert more influence. $Z_i = \sum_{j \in \mathcal{N}(i)} w_{ij}$ normalizes by total neighbor weight. The second term rewards contrast with non-neighbors: candidates are penalized for collapsing toward descriptions of structurally unrelated inputs. λ controls the relative strength of this contrastive term.

We compare structured descriptions using normalized edit distance (NED), a Levenshtein-style distance over tuple sequences (Levenshtein, 1966). Insertions and deletions cost 1, while substitutions receive partial credit for matching tuple fields (e.g., `(rise, gradual)` → `(rise, sharp)` costs 0.5). We normalize by the longer sequence length to compare descriptions of different lengths. For each input x_i, we select the highest-scoring candidate as the chosen response and the lowest-scoring candidate as the rejected response. For quality control purposes, we discard anchors whose chosen and rejected candidates have a score difference below a fixed threshold to limit their weak-preference supervision effects.

$$y_i^+ = \arg\max_{c \in \mathcal{C}_i} S(c \mid x_i), \qquad y_i^- = \arg\min_{c \in \mathcal{C}_i} S(c \mid x_i).$$

Now, repeated for many inputs $\{x_1, x_2, \ldots x_N\}$ we construct a training dataset $\{(x_1, y_1^+, y_1^-), (x_2, y_2^+, y_2^-), \ldots (x_N, y_N^+, y_N^-)\}$.

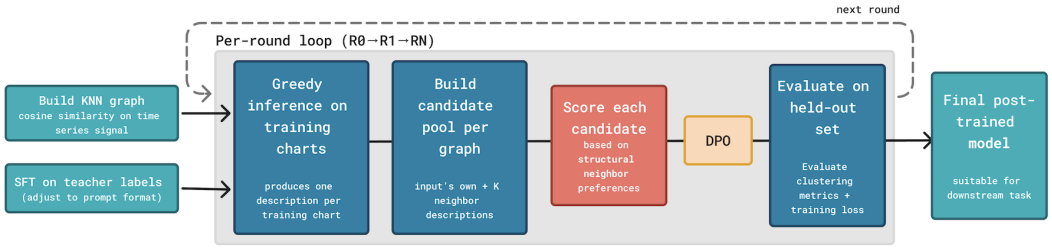


Figure 2: Domain-agnostic iterative training pipeline to evaluate our preference learning mechanism. Preference pairs are derived purely from structural relationships among unlabeled inputs, with zero task supervision at any stage.

3 Training Pipeline

Section 2 introduced a mechanism for constructing preference pairs from structural relationships among unlabeled inputs. To test whether the signal improves alignment between visual and textual descriptions, we instantiate it using a general post-training pipeline. The pipeline is designed to be domain-agnostic: it accepts any collection of visual inputs, applies structural-neighbor preference construction, and evaluates the trained model on zero-shot downstream tasks. The pipeline has three parts: (1) SFT initialization so the model learns the output format, (2) structural-neighbor preference pair construction, and (3) DPO update and zero-shot evaluation on held-out downstream tasks. No human labels or task-specific supervision are used in preference construction or post-training.

SFT Initialization. We first fine-tune the model on a small set of structured descriptions generated by a teacher model, which is a separate, more capable vision-language model prompted to produce outputs in the tuple format described in Section 2.1. This stage teaches the base model the structured tuple vocabulary and syntax required for edit-distance scoring, but provides no structural-neighbor supervision and is not used to construct preference pairs.

DPO Update and Iteration. We run $R = 3$ rounds of iterative DPO. In each round r, the current policy model generates a description for every training example, and the fixed-neighbor graph is used to construct candidate pools and score them from the scoring function, yielding a preference triple (x_i, y_i^+, y_i^-) for each training example. We set $R = 3$ for all experiments, chosen due to compute availability.

The current model R_r is kept fixed as the reference model, and the newly trained LoRA (Hu et al., 2022) adapter serves as the labeling model in the next round R_{r+1}. We include an iterative DPO pipeline because we recognize that a single DPO round may be limited by the quality of the initial descriptions used to construct preference pairs. As the model improves through rounds, its descriptions (candidates) become more reliable, enabling higher-quality preference pairs in subsequent rounds.

Final Policy and Held out Evaluation. The adapter after the final DPO round is used for downstream evaluation. We evaluate it zero-shot on held-out tasks to test whether structural-neighbor supervision improves visual-text alignment and transfers to tasks the model never trains on.

4 Results

We instantiate the structural-neighbor preference signal through DPO and apply it across three unrelated time series domains: financial price charts (Center for Research in Security Prices, 2025), ECG waveforms (Wagner et al., 2022), and seismic traces (Swiss Seismological Service (SED) at ETH Zürich, 1983). The base model is Qwen2-VL-7B Instruct (Wang et al., 2024b), and we replicate experiments on Gemma-3-4B (Gemma Team, 2025). Each domain constructs structural neighborhoods from the raw underlying signal: for finance, the raw price path, for ECG median-beat Lead-II waveforms, and for seismology broadband traces. $K = 10$ nearest neighbors, and LoRA adapters are trained with $\beta = 0.3$, learning rate 2×10^{-6}, and three epochs per round. From a sensitivity sweep, λ was chosen as 0.3, $Y = 2$, and the similarity threshold is 0.15.

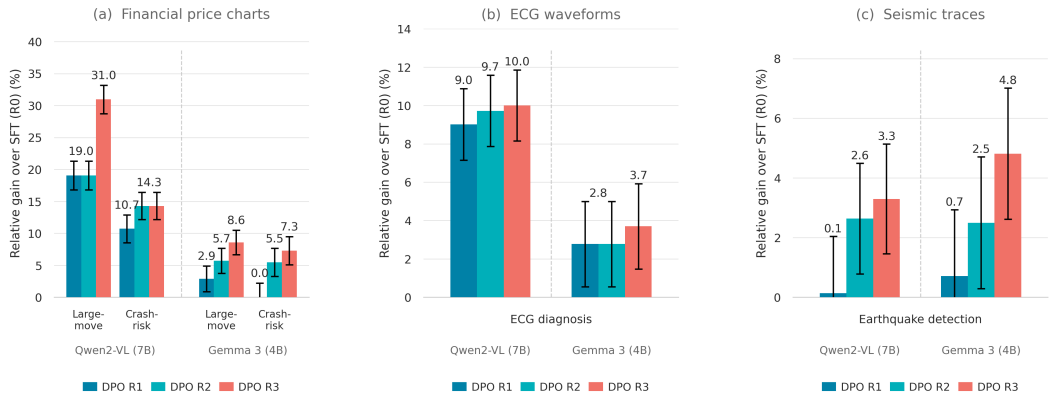


Figure 3: Relative gain over SFT (%) across 3 rounds of iterative DPO on financial, ECG, and seismic domains. Zero-shot downstream performance improves monotonically across rounds with no task-specific supervision at any stage. (a) and (b) demonstrate statistically significant improvements.

We evaluate whether structural-neighbor preference learning transfers zero-shot to downstream tasks that the model never trains on. For each domain, we prompt the post-trained model with a chart image and a domain-specific question (see Figure 1 for shortened versions of the questions). We then run mechanistic experiments (Qwen2-VL reported) to study internal model changes, and ablations to confirm gains are specific to structural-neighbor supervision rather than generic preference optimization. Together, these measures converge on the conclusion that structural-neighbor preference learning produces genuine visual grounding improvement.

4.1 Zero-shot downstream transfer across three time-series domains.

We find that the structural-neighbor signal produces zero-shot improvements on independently-labeled benchmarks in all three domains on tasks the model was never trained for. Across the domains, gains are realized without domain-specific labels or downstream task supervision at any stage, using preference-pairs constructed from raw-signal structural neighborhoods. All results below use seed 42; across three full training seeds, mean gains are within 0.8 points, see Table 2.

Financial Price Charts. We evaluate two tasks: whether the models can predict a large price movement and whether there is an imminent crash. Large-move detection asks the model whether the chart predicts unusually high future movement after the time series, and crash-risk detection asks if the chart contains distress-like structure that predicts future market decline. After three rounds, Qwen2-VL achieves a 31.0% and 14.3% relative gain over SFT on large-move and crash-risk detection respectively, while Gemma 3 (4B) improves by 8.6% and 7.3%. These gains (statistically significant) arise without any exposure to volatility or crash-risk labels, indicating that structural-neighbor training improves sensitivity to price-path geometry relevant for financial risk interpretation.

ECG Waveforms. We evaluate whether the descriptions the model learns on its own, with no clinical labels, can better separate healthy from abnormal ECG waveforms. We score each ECG by how cleanly the model can separate ECG waveforms based on abnormality. In this case, the model is never informed what an abnormal vs. normal ECG looks like, and is able to achieve a 10.0% relative gain over SFT after three rounds on Qwen2-VL and 3.7% on Gemma 3 (4B).

Seismic Traces. We evaluate whether the model learns to separate earthquake waveforms from noise, requiring sensitivity to high-frequency structure it receives no supervision for. Qwen2-VL achieves a 3.3% relative gain over SFT after three rounds, while Gemma 3 (4B) reaches 4.8%. Gains accumulate across rounds in both models, indicating that the structural-neighbor signal progressively sharpens sensitivity to the fine-grained waveform features that distinguish seismic events from noise.

Comparison to preference learning baselines: To test whether the gains reflect structural-neighbor supervision rather than preference optimization, we compare against image-conditioned, self-rewarding, and external-judge DPO baselines on the financial tasks (Wang et al., 2024a; Yuan et al., 2024; Ouali et al., 2024; Zhai et al., 2023). All methods start from the same SFT checkpoint

Preference-learning baselines

Method	Crash ↑	Move ↑
Ours	**+14.3%**	**+31.0%**
SigLIP-DPO	+2.9%	+26.7%
CLIP-DPO	+5.9%	+25.7%
mDPO	+1.4%	+24.5%
Self-rewarding	+4.5%	+23.6%
AI judge	+2.9%	+22.4%

Neighborhood source

Source	Crash ↑	Move ↑
Frozen encoder	0.0%	0.0%
DINOv2	+6.3%	+21.4%
Image trajectory	+11.6%	+23.8%
Raw time series	**+14.3%**	**+31.0%**

Table 1: Preference-learning baselines (left) and neighborhood sources (right), reported as relative improvement over SFT (%) on financial tasks.

and use the same training examples, number of preference pairs, and evaluation protocol. Structural neighbor preference learning achieves the strongest gains on both tasks.

4.2 Preference learning improves cross-modal alignment

Having shown zero-shot transfer to tasks for which the model has not been trained, we investigate whether our preference learning mechanism promotes stronger cross-modal alignment in foundation models. After training with structural-neighbor preference pairs, the model exhibits stronger cross-modal alignment: its textual outputs are more faithful to the visual inputs and structure. We measure alignment as the Spearman correlation between CLIP (Radford et al., 2021) visual similarity and description similarity over 20,000 chart pairs. From SFT to after three DPO rounds, alignment rises by 54% ($p < 0.000001$), as shown in Figure 4b, with similar gains of 49 - 55% using alternative measures SigLIP (Zhai et al., 2023), DINOv2 (Oquab et al., 2024), and dynamic time warping over image-extracted trajectories. The distance among visually similar chart pairs falls from 2.155 to 1.858, while descriptions of non-similar charts remain well separated (2.383). This indicates that the model is not just producing better clustered outputs, but descriptions are becoming more similar when underlying visual perception is similar. Since CLIP is never used during training, this alignment gain cannot reflect direct optimization pressure, and it provides independent evidence that structural-neighbor supervision reorganizes the model's description space around visual structure.

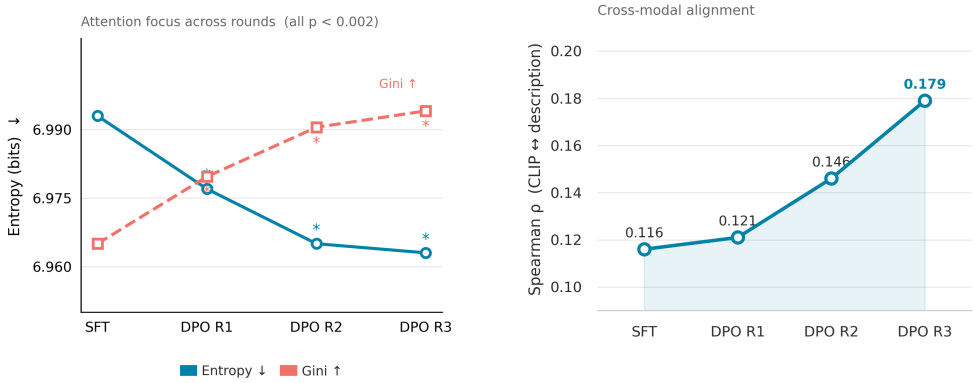
Alignment by itself, however, does not establish correctness, and we therefore evaluate about 1,500 preference pairs from CUB (Wah et al., 2011) using human-annotated bird attributes. The preferred description matches the annotations better than the rejected description in 60.1% of pairs, compared with 48.0% under random assignment. This indicates the preference signal tends toward more visually grounded descriptions, though errors shared across neighborhoods can stay undetected.

4.3 Gains are evident throughout the network

Cross modal alignment shows that the model's descriptions become more faithful to independent visual similarity, and we next ask where this change appears in the model. We conduct three analyses: hidden state alignment to see which layers shift, attention selectivity to investigate concentration of model's focus around the chart, and neighborhood precision to more directly test the structural-preference learning pairs cause similar examples to end up in the same families.

Hidden State Alignment. We measure the Spearman correlation ρ between LLM hidden-state cosine similarity and CLIP cosine similarity for 5000 held-out chart pairs. Effectively, this measures whether, within layers, the model's internal representations become more visually grounded. Figure 5a shows alignment increases at every measured depth from SFT to R3, with all correlations significant under Bonferroni correction. The strongest alignment appears at layer 20, suggesting that visual structure may be captured most clearly in the middle-to-late layers before being reshaped for a textual output during decoding. We also see considerably higher relative gains earlier at layer six, doubling from 0.059 to 0.116, suggesting that the structural-preference learning mechanism changes a model's internal representation of visual structure throughout the network.

Attention Selectivity. We examine whether the model's focus on a chart image changes due to preference-pair supervision. Attention over image tokens becomes more concentrated, with entropy decreasing from 6.993 to 6.963 from SFT to R3 and Gini increasing (Figure 4a), both changes



(a) Image-token attention entropy decreases and Gini coefficient increases across DPO rounds ($p < 0.002$), indicating that the model attends more selectively to visual structure over training.

(b) Cross-modal alignment increases 54% ($p < 0.000001$) over SFT across iterative DPO rounds, indicating increasing alignment between the model's vision and textual modalities.

Figure 4: Structural-neighbor preference training strengthens visual grounding across measures.

statistically significant under paired t-test and Wilcoxon signed-rank tests. This indicates the model doesn't spread attention equally on the whole chart, but to particular chart dynamics.

The effect is strongest on texture tokens, where entropy drops by 0.044, compared with only a 0.014 drop for shape-describing (state) tokens. Lower entropy indicates model attention is less spread out, concentrated on a smaller region in the image. The decrease in the texture domain shows structural-neighbor preference learning sharpens local distinctions between similar signals, as texture tokens generally capture fine-grained shape properties as compared to directional state tokens.

Neighborhood Precision. For each chart, we retrieve its top-K visual neighbors using CLIP and measure how often those neighbors fall into the same family using traditional clustering techniques, i.e. HDBSCAN (McInnes et al., 2017). Figure 5b shows precision rises 21% from SFT to R3, indicating that the model's description space is more locally consistent with visual structure. The structural neighbor preference learning claim is built on the idea that similar inputs should receive related descriptions, and neighborhood precision tests evaluate this at the example level. By the final round, we see visually similar charts are more likely to land in the same (originally clustered) description family, aligning with local structure that the preference learning signal is meant to induce.

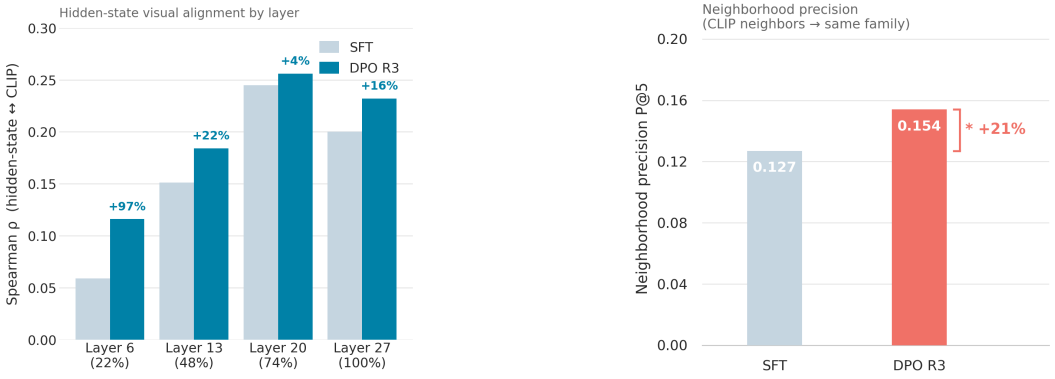
4.4 Extending Preference Learning Beyond Time-Series

Generalization Beyond Time Series. We apply this preference learning pipeline to histograms, scatterplots, and natural images, obtaining relative gains over SFT of 35.2%, 20.0%, and 18.8%, respectively. For our natural images (CUB bird images), neighborhoods are constructed directly from DINOv2 image embeddings, with no corresponding raw signal, showing that the method is not restricted to underlying data. These results suggest that structural-neighbor supervision is not specific to time-series data, but extends to other visual domains where the neighborhood representation contains meaningful structure.

Image-derived neighborhoods. We reconstruct the financial neighbor graph using only chart images, using either DINOv2 embeddings or trajectories extracted from chart pixels. Both improve downstream performance, with extracted trajectories recovering 81% of the crash-risk gain and 77% of the large-move gain from raw-signal neighborhoods (Table 1). Thus, the method depends on preserving task-relevant structure, not access to the original signal.

4.5 Ablations

We ablate components central to structural-neighbor preference signal to isolate what drives downstream transfer. Each ablation modifies a single component while holding the rest of the pipeline fixed. Full results are reported in Appendix A.6.



(a) Hidden-state similarity becomes more aligned with visual similarity ($p < 10^{-14}$ under Bonferroni).

(b) Visual neighbors increasingly map to the same description family ($p < 0.001$).

Figure 5: Representation-level visual alignment improves after preference training.

Prompt and output format. We test whether gains depend on hand-engineered prompts or constrained output vocabulary by varying both prompt specificity and response format. We test on structured tuple outputs, simpler single-state outputs, general prompts, LLM-generated prompts, and unrestricted free text. Across general prompts, LLM-generated prompts, and free text, all variants improve over SFT with large-move gains from 28.3–30.6% compared with 31.0% using the curated format, showing that the preference signal does not depend on a particular prompt or fixed vocabulary. See Tables 10 and 11.

Scoring ablation. Replacing NED with raw edit distance introduces a length bias, favoring verbose descriptions over structurally accurate ones and corrupting the preference pairs. Crash-risk prediction drops to near the SFT baseline, confirming that normalization is necessary for the preference signal to reflect structural similarity rather than output length.

Training procedure. We compare DPO against two iterative SFT variants. Standard iterative SFT overfits and produces degenerate `[ ( [ ( [ (` loops at inference. Extending LoRA to the vision encoder fails to converge and yields no valid outputs across 800 generations. The first failure suggests preference supervision is needed for the model to learn distinctions between inputs, rather than only imitating output format. The vision-encoder failure shows that updating a pretrained visual representation can destabilize training. MMLU (Hendrycks et al., 2021) performance remains unchanged across DPO rounds, showing no degradation in general reasoning (A.8).

5 Discussion

Structural relationships among unlabeled inputs carry a grounding signal strong enough to improve vision-language models on tasks they never train on. Applying structural-neighbor preference learning to Qwen2-VL-7B and Gemma-3-4B increases performance on zero-shot transfer tasks across unrelated time series domains, histograms, scatterplots, and natural images. These downstream tasks were never seen during training. Mechanistic experiments provide evidence of reorganization in the model space: across several independent tests, we observe statistically significant changes in attention and hidden states, with effects appearing throughout early, middle, and later layers of the model. Cross-modal alignment between descriptions and corresponding images rises substantially across DPO rounds, indicating that the model's language is becoming more tightly coupled to what it actually sees. The ability to replicate this behavior across model families suggests that the improvements are not architecture dependent. The results provide evidence that the structural-neighbor preference learning mechanism does more than teach the model to write better descriptions: it changes how visual structure is represented and reflected in model outputs. Ablations show that performance depends on preserving task-relevant structure, rather than access to the original signal or a particular output format. Image-derived neighborhoods retain substantial gains, while structured tuples, single-state outputs, and free text all improve over SFT.

Broader Impact. Adapting multimodal foundation models to specialized domains often requires preference data that is difficult to obtain in those domains. Structural-neighbor preference learning instead turns structure already present in unlabeled data into a training signal, reducing reliance on human preference labels, task-specific supervision, and external judges. This may make multimodal foundation-model post-training more practical in scientific and specialized domains where unlabeled data are abundant but high-quality supervision is limited.

Limitations. Structural-neighbor preference learning requires a similarity function that preserves task-relevant structure. While raw signals provide the strongest neighborhoods in our time-series experiments, image-derived representations can work if they preserve sufficient neighborhood quality. Our structured time-series prompts also encode some design choices, such as reading charts from left to right, although our prompt and free-text ablations show that these choices are not required for the preference signal to remain effective. We have tested the mechanism only with DPO and only on smaller-scale models due to compute budgets. Each model is also trained on one domain at a time, and whether structural-neighbor signals from multiple domains can be fused is an open question.

6 Related Work

6.1 Multimodal Preference Optimization and Visual Grounding

Direct Preference Optimization (DPO; Rafailov et al., 2023)is widely used for post-training multimodal foundation models, with prior work focused on building preference pairs that improve response quality and visual grounding (Zhou et al., 2024a; Yu et al., 2024). These preference signals can come from human annotations, model-based scores, image-text similarity, or synthetic changes such as hallucination injection and image corruption (Li et al., 2023; Ouali et al., 2024; Zhou et al., 2024a). Iterative and self-rewarding methods instead use the model's own generations or feedback to update preference signals across rounds (Yuan et al., 2024; Chen et al., 2024; Zhou et al., 2024b).

A known issue is the unconditional preference problem, when a model learns to prefer better-sounding text without relying equally on the image (Wang et al., 2024a). mDPO addresses this with image-aware preferences and reward anchoring, while other work uses vision-guided contrasts based on modified versions of the same image (Xie et al., 2024), including fully self-supervised variants (Zhu et al., 2024). These methods make the preference signal more image-aware, but still build it within a single image-response example. We instead ask whether relationships across structurally similar inputs can directly provide the preference signal.

6.2 Structure-derived and self-supervised preference signals

Our method ties into a line of work that uses structures in data as supervision. Contrastive methods pull similar examples together and push distinct ones apart without labels (Chen et al., 2020; He et al., 2020), and iterative alignment methods use model-generated responses or evaluations to refine outputs (Yuan et al., 2024; Chen et al., 2024). More recent work incorporates harder negatives and retrieval to make preference signals sensitive to visual evidence (Xing et al., 2025).

7 Conclusion

We introduce structural-neighbor preference learning, a self-supervised mechanism for constructing preference pairs from structural relationships among unlabeled inputs. The central result is that structural relationships between similar inputs carry a grounding signal strong enough to improve vision-language models through preference optimization alone. Applied across three unrelated time-series domains and replicated on two model families, the method produces consistent zero-shot transfer to held-out tasks, supported by mechanistic evidence of internal model reorganization toward visual structure. Because preference-pair construction requires only a meaningful similarity function over inputs, the method can operate wherever useful structural relationships exist among unlabeled data. We evaluate the resulting preference signal with DPO here, and testing other preference-learning objectives remains future work. Because the method requires only a meaningful similarity function over inputs and is agnostic to model architecture, it applies broadly wherever structured relationships exist among unlabeled data.

AI USE STATEMENT

In this work, we used generative AI tools for drafting some code throughout this research (creating and editing code, implement methods, debugging), producing code for files that was then reviewed by a human reader. This also includes polishing text and improving writing. We have not used generative AI tools for research ideation. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses existing datasets and does not involve new data collection or human-subject experiments. We use financial, ECG, and seismic datasets as research settings for evaluating multimodal grounding and post-training methods.

REPRODUCIBILITY STATEMENT

Section 2 specifies the preference-construction procedure, including neighborhood construction, scoring, and normalized edit distance, while Section 3 describes the SFT and iterative DPO pipeline. Hyperparameters, LoRA configuration, compute, and seeding are reported in Section 4 and Appendix A.2.7, with three-seed results in Appendix A.2.8. Appendix A.2 provides all prompts and output vocabularies, Appendix A.1 example signals; ablation details appear in Appendix sections A.6, A.7.1, and A.7.3. PTB-XL, CUB-200-2011, and Swiss Seismological Service data are publicly available.

REFERENCES

Center for Research in Security Prices. CRSP Daily Stock File. Wharton Research Data Services, 2025. Accessed through WRDS. Center for Research in Security Prices, University of Chicago Booth School of Business.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the International Conference on Machine Learning*, 2020.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*, 2024.

Gemma Team. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

Lei Li, Zhihui Xie, Mukai Li, Shunian Chen, Peiyi Wang, Liang Chen, Yazheng Yang, Benyou Wang, and Lingpeng Kong. Silkie: Preference distillation for large visual language models. *arXiv preprint arXiv:2312.10665*, 2023. URL https://arxiv.org/abs/2312.10665.

Leland McInnes, John Healy, and Steve Astels. hdbscan: Hierarchical density based clustering. *J. Open Source Softw.*, 2(11):205, 2017.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. URL https://arxiv.org/abs/2304.07193.

Yassine Ouali, Adrian Bulat, Brais Martinez, and Georgios Tzimiropoulos. Clip-dpo: Vision-language models as a source of preference for fixing hallucinations in lvlms, 2024. URL https://arxiv.org/abs/2408.10433.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

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.

Swiss Seismological Service (SED) at ETH Zürich. National seismic networks of switzerland. ETH Zürich, 1983.

Patrick Wagner, Nils Strodthoff, Ralf-Dieter Bousseljot, Wojciech Samek, and Tobias Schaeffter. PTB-XL, a large publicly available electrocardiography dataset. *PhysioNet*, November 2022. doi: 10.13026/kfzx-aw45. URL https://doi.org/10.13026/kfzx-aw45. Version 1.0.3.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. Caltech-ucsd birds-200-2011. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Fei Wang, Wenxuan Zhou, James Y. Huang, Nan Xu, Sheng Zhang, Hoifung Poon, and Muhao Chen. mdpo: Conditional preference optimization for multimodal large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2024a.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024b.

Yuxi Xie, Guanzhen Li, Xiao Xu, and Min-Yen Kan. V-dpo: Mitigating hallucination in large vision language models via vision-guided direct preference optimization. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 13258–13273, 2024.

Shuo Xing, Peiran Li, Yuping Wang, Ruizheng Bai, Yueqi Wang, Chan-Wei Hu, Chengxuan Qian, Huaxiu Yao, and Zhengzhong Tu. Re-align: Aligning vision language models via retrieval-augmented direct preference optimization. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 2379–2397. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.121. URL https://aclanthology.org/2025.emnlp-main.121/.

Tianyu Yu, Yuan Yao, Haoye Zhang, Taiwen He, Yifeng Han, Ganqu Cui, Jinyi Hu, Zhiyuan Liu, Hai-Tao Zheng, Maosong Sun, and Tat-Seng Chua. RLHF-V: Towards trustworthy MLLMs via behavior alignment from fine-grained correctional human feedback. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason E. Weston. Self-rewarding language models. *ArXiv*, abs/2401.10020, 2024. URL https://arxiv.org/abs/2401.10020.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training, 2023. URL https://arxiv.org/abs/2303.15343.

Yiyang Zhou, Chenhang Cui, Rafael Rafailov, Chelsea Finn, and Huaxiu Yao. Aligning modalities in vision large language models via preference fine-tuning, 2024a. URL https://arxiv.org/abs/2402.11411.

Yiyang Zhou, Zhiyuan Fan, Dongjie Cheng, Sihan Yang, Zhaorun Chen, Chenhang Cui, Xiyao Wang, Yun Li, Linjun Zhang, and Huaxiu Yao. Calibrated self-rewarding vision language models. *arXiv preprint arXiv:2405.14622*, 2024b.

Ke Zhu, Zheng Ge, Liang Zhao, and Xiangyu Zhang. Self-supervised visual preference alignment, 2024. URL https://arxiv.org/abs/2404.10501.

A Appendix

A.1 Time Series Signals

A.1.1 Financial Price Series

An example of a financial price series:

```
[1.0000, 0.9940, 1.0050, 1.0180, 1.0440, 1.0790, 1.0720, 1.0600,
1.0310, 1.0540, 1.0480, 1.0430, 1.0650, 1.1080, 1.1310, 1.1200,
1.1130, 1.1090, 1.1100, 1.1090]
```

A.1.2 ECG Waveform

An example of ECG Waveform (a normal sinus rhythm):

```
[-0.02, -0.02, -0.01, -0.02, -0.02, -0.01, -0.03, 0.06, 0.14,
0.22, 0.06, -0.08, -0.12, -0.08, -0.01, 0.00, 0.00, 0.01, 0.00,
-0.01, -0.48, 0.65, 0.49, 0.12, -0.10, -0.08, -0.06, -0.05, -0.05,
-0.06, -0.06, -0.06, -0.05, -0.05, -0.06, 0.00, 0.03, 0.05, 0.07,
0.09, 0.10, 0.09, 0.08, 0.07, 0.05, 0.03, 0.01, -0.01]
```

A.1.3 Seismic Trace

An example of an earthquake with clear P-wave onset:

```
[0.02, -0.01, 0.03, -0.02, 0.01, 0.00, 0.02, -0.01, 0.03, -0.02,
0.01, 0.00, 0.02, -0.01, 0.03, -0.02, 0.01, 0.00, 0.02, -0.01,
0.03, -0.02, 0.01, 0.00, 0.52, 1.21, 1.54, 0.98, 0.21, -0.63,
-1.42, -1.89, -1.21, -0.34, 0.48, 1.13, 1.28, 0.82, 0.14, -0.52,
-1.01, -0.98, -0.52, 0.08, 0.61, 0.79, 0.56, 0.18, -0.28, -0.75,
-1.05, -0.82, -0.31, 0.22, 0.58, 0.57, 0.31, 0.04, -0.18, -0.42]
```

A.2 Prompts

We use domain-specific structured prompts for each time-series domain. Each prompt constrains the model to produce an ordered list of tuples using a fixed vocabulary, enabling reliable edit-distance scoring.

A.2.1 Financial Price Charts

```
You are labeling a normalized price chart from left to right.

Task:
Output an ordered list of tuples describing the main visual phases of
the chart.

Important:
- Start at the far left of the chart.
- End at the far right of the chart.
- The first tuple must describe the first major region at the left
```

```
edge.
- Each tuple must describe one contiguous region of the chart.
- Choose the number of tuples based on the chart.
- Use only the states and textures that clearly fit the chart.
- Describe the whole chart at a high level.
- Ignore small wiggles, tiny spikes, and one-step reversals unless
they clearly define a major region.

State meanings:
- rise = a region where the line mostly moves upward from left to
right
- fall = a region where the line mostly moves downward from left to
right
- sideways = a region where the line stays around a similar level
overall
- peak = a noticeable top region where the line reaches a high area
and then turns downward
- trough = a noticeable bottom region where the line reaches a low
area and then turns upward
- breakout = a strong upward move that begins after an earlier flat or
range-like region
- breakdown = a strong downward move that begins after an earlier flat
or range-like region

Texture meanings:
- gradual = the line changes smoothly over the region, without sudden
jumps
- sharp = the line changes suddenly over the region, with a steep move
or abrupt turn
- choppy = the line moves up and down repeatedly, with several small
reversals
- flat = the line stays almost horizontal, with very little movement

Allowed states:  rise, fall, sideways, peak, trough, breakout,
breakdown
Allowed textures:  gradual, sharp, choppy, flat

Output only the tuple list.
```

A.2.2 ECG WAVEFORMS

```
Label this ECG waveform from left to right as an ordered list of
tuples.

Each tuple must be one of these patterns.  Pick the one that best
matches each region:

(flat, smooth) --- clean baseline staying at one level
(flat, choppy) --- jittery / noisy baseline with small up-and-down
wiggles
(flat, elevated) --- flat segment sitting visibly above the
surrounding baseline
(flat, depressed) --- flat segment sitting visibly below the
surrounding baseline
(flat, low_amplitude) --- whole waveform is compressed vertically.
Use at most once.

(peak, small) --- small rounded bump (e.g. a P wave)
(peak, gradual) --- wider rounded bump (e.g. a typical T wave)
(peak, pointed) --- bump that comes to a sharp tip instead of a round
top

(trough, sharp) --- small / shallow downward dip
(trough, deep) --- large downward dip, much lower than other troughs
(trough, gradual) --- wider rounded dip (e.g. an inverted T wave)

(spike, tall) --- tall thin upward protrusion (the QRS R peak)
```

```
(spike, wide) --- tall protrusion visibly broader than a thin spike
(spike, notched) --- tall protrusion with a secondary bump on its rise
or fall
(rise, gradual) --- line moves upward smoothly across the region
(fall, gradual) --- line moves downward smoothly across the region
Rules:
- Output 3 to 10 tuples covering the waveform left to right.
- Judge wide / deep / elevated / depressed relative to the rest of the
same waveform.
- Two waveforms that look different should produce different tuple
lists.
Output only the tuple list.
```

A.2.3 SEISMIC TRACES

```
You are describing a chart with a wavy line moving from left to right.
Output an ordered list of pairs:  (shape, line_quality)

SHAPE --- pick exactly one:
flat --- the line stays at about the same level
rise --- the line moves upward overall
fall --- the line moves downward overall
peak --- a high turning region where the line changes from rising to
falling
trough --- a low turning region where the line changes from falling to
rising
wavy --- repeated or dense small up-and-down movement with no clear
overall direction
spike --- a single sudden narrow jump or drop that sharply stands out
LINE_QUALITY --- pick exactly one:
clean --- smooth curve with little or no visible jitter
fuzzy --- a larger shape is visible, but small fast wiggles ride on
top
chaotic --- dominated by sharp irregular jumps, with no clear smooth
shape underneath
Rules:
- Start at the far left and end at the far right.
- Cover every part of the chart exactly once.
- Write pairs in left-to-right order.
- Adjacent pairs must differ in at least one field.
- Do not create new pairs for tiny wiggles unless the larger visible
behavior changes.
Output only the list of pairs.  No explanation, no preamble.
```

A.2.4 HISTOGRAMS

```
You are describing the bar heights of a histogram from left to right.
Output an ordered list of pairs:  (state, texture)

STATE --- pick exactly one:
climb --- heights rise across the region with no peak at its end
decline --- heights fall across the region with no peak at its start
plateau --- heights stay at a similar level with no clear trend
mode --- a peak clearly taller than the bars on either side
valley --- a dip clearly shorter than the regions on both sides
tail --- a long low run at the far left or far right, well below the
```

```
tallest bars
shoulder --- a raised run beside a taller mode, above the surrounding
level


TEXTURE --- pick exactly one:
smooth --- heights change gradually from bar to bar
ragged --- heights jump irregularly between neighboring bars
steep --- the region rises or falls sharply over only a few bars
gentle --- the region rises or falls slowly over many bars
narrow --- the region spans only a few bars
wide --- the region spans many bars
heavy --- the bars are tall relative to the rest of the chart
light --- the bars are short relative to the rest of the chart


Rules:
- Start at the far left and end at the far right.
- Cover every bar exactly once; each region spans at least two bars.
- Start a new pair only when the state or the texture clearly changes.
- Judge heights by what you see, not by the y-axis numbers.
- Use tail only at an edge, and shoulder only when a taller mode
exists elsewhere.
- Do not create new pairs for single tall or empty bars; fold them
into their region.


Output only the list of pairs.  No explanation, no preamble.
```

A.2.5 SCATTER PLOTS

```
You are describing a scatter plot as a sequence of regions read from
left to right.
Output one line in the form:  Reasoning:  [(pattern, position), ...]


PATTERN --- pick exactly one:
corr_positive --- points trend upward as x increases
corr_negative --- points trend downward as x increases
corr_none --- no clear trend between x and y
corr_nonlinear --- a clear trend that bends or changes direction
shape_linear --- points fall along a narrow straight line
shape_curved --- points fall along a bent or arced path
shape_blob --- a roughly round cloud with no elongation
shape_band --- a wide elongated strip of points
outliers_isolated --- stray points far from every cluster


POSITION --- pick exactly one:
skew_upperright, skew_upperleft, skew_lowerright, skew_lowerleft ---
the region's centre sits in that quadrant of the plot
skew_balanced --- the region's centre sits near the middle
few_outliers, many_outliers --- only for the outlier pair


Rules:
- One pair per visible cluster, ordered left to right along the
x-axis.
- If stray points sit apart from every cluster, add one outlier pair
last.
- So k clusters give k pairs, and k clusters with outliers give k+1.
- Use outliers_isolated and few/many_outliers only in that final pair.
- Do not create a cluster that is not visibly separated from its
neighbors.
```

```
Output only the Reasoning line.  No explanation, no preamble.
```

A.2.6 NATURAL BIRD PHOTOGRAPHS (CUB-200-2011)

```
You are labeling the visible physical attributes of the bird in this
photograph.
Output an ordered list of pairs in the form:  [(part, value), ...]

PART --- pick from this list only, and take the value from the
matching set:
eye_color, throat_color, crown_color, forehead_color, nape_color,
breast_color, belly_color, underparts_color, upperparts_color,
primary_color, bill_color --- COLORS
breast_pattern, belly_pattern --- PATTERNS
bill_length --- LENGTHS

VALUE --- pick exactly one, verbatim from the CUB-200-2011 vocabulary:
COLORS --- black, blue, brown, buff, green, grey, iridescent, olive,
orange,
pink, purple, red, rufous, white, yellow
(eye_color is the one exception:  iridescent is not allowed for it)
PATTERNS --- solid, spotted, striped, multi-colored
LENGTHS --- shorter_than_head, about_the_same_as_head, longer_than_head

Rules:
- Include a part only if it is clearly visible in the photograph.
- Always include eye_color, throat_color and crown_color.
- Do not repeat a part.

Example:  [(eye_color, black), (throat_color, black)]

Output only the list of pairs.  No explanation, no preamble.
```

A.2.7 COMPUTE RESOURCES, WALL TIME, AND REPRODUCIBILITY

Computational Resources and Reproducibility. All experiments were run on a single NVIDIA L40S GPU (48 GB VRAM) with 128 GB host RAM and 4 CPU cores. The base model is Qwen2-VL-7B-Instruct fine-tuned via LoRA (rank 16, $\alpha = 32$, dropout 0.05) targeting the query, key, value, and output projection layers. Each DPO round trains for 3 epochs with a batch size of 1 and gradient accumulation of 4 (effective batch 4), learning rate 2×10^{-6}, warmup ratio 0.1, and DPO $\beta = 0.3$. Training a single round takes approximately 2 hours 30–2 hours 45 minutes depending on the number of training examples. For reproducibility, our reported results use a seed of 42; HuggingFace set_seed is used to seed PyTorch, NumPy, and Python's random module jointly. We run three separate seeds; all numbers reported elsewhere in the paper come from the seed-42 run, and Table 2 reports the mean across the three seeds.

A.2.8 MULTI-SEED ROBUSTNESS

Because DPO and LoRA training can be sensitive to initialization, we re-ran the full three-round pipeline end to end on three random seeds in each domain. Table 2 reports the mean relative gain over SFT across seeds alongside the seed-42 run used throughout the paper. The mean over three seeds is within 0.8 points of the seed-42 value on every task, and the ordering of gains across domains is unchanged.

Table 2: Relative gain over SFT (%) on Qwen2-VL across three training seeds. All other tables report the seed-42 run; this table additionally reports the mean across seeds.

Domain	Held-out task	Seed 42 (reported)	Mean over 3 seeds
Finance	Large-movement detection	31.0	31.8
Finance	Crash-risk detection	14.3	14.1
ECG	Abnormality detection	10.0	10.8
Seismic	Earthquake detection	3.3	3.4

A.2.9 TEXT CLUSTERING QUALITY

Table 3 gives clustering metrics on 400 held-out eval charts across three DPO rounds. HDBSCAN hyperparameters are fixed (*min_cluster_size*=4).

Table 3: Text clustering metrics. Sep. Ratio = between-family ED / within-family ED. ARI between adjacent rounds: R1 vs. R2 = 0.424; R2 vs. R3 = 0.459.

Checkpoint	Sep. Ratio ↑	Silhouette ↑	Families
SFT (R0)	3.60	0.297	80
DPO R1	4.53	0.361	80
DPO R2	7.83	0.531	80
DPO R3 (largest)	**9.96**	**0.571**	80

A.2.10 ATTENTION ENTROPY ANALYSIS

Table 4 reports full attention statistics from the last four transformer layers for $n = 20$ eval charts, comparing each DPO checkpoint against the SFT baseline via paired t-test and Wilcoxon signed-rank test.

Table 4: Attention entropy and Gini coefficient (SFT vs. each DPO round). All p-values confirmed by Wilcoxon signed-rank ($W = 13$–17, $p < 0.002$).

Round	SFT Entropy	DPO Entropy	Δ Entropy	Paired t	p-value	Gini Δ
R1	6.993 ± 0.044	6.977 ± 0.050	-0.016	4.034	0.0007^{***}	$+0.0049^{***}$
R2	6.993 ± 0.044	6.965 ± 0.049	-0.028	3.739	0.0014^{**}	$+0.0085^{***}$
R3	6.993 ± 0.044	6.963 ± 0.053	-0.031	4.045	0.0007^{***}	$+0.0097^{***}$

Table 5 shows per-token-type entropy reduction at R3 vs. SFT. Texture tokens (e.g., `choppy`, `gradual`, `sharp`) drive the largest reduction, consistent with the model learning to distinguish fine-grained visual texture rather than coarse directional structure.

A.2.11 CROSS-MODAL ALIGNMENT

Table 6 reports Spearman ρ between CLIP cosine similarity and description similarity derived from raw edit distance over 20,000 random chart pairs per round. CLIP embeddings are never used during training. The ED columns report raw edit distance rather than the normalized edit distance (NED) used for preference-pair scoring.

A.2.12 LAYER-WISE HIDDEN-STATE ALIGNMENT

Table 7 reports Spearman ρ between CLIP cosine similarity and LLM hidden-state cosine similarity at four network depths, computed over 5,000 random pairs from 241 eval charts. All values are significant at $p < 10^{-14}$ (Bonferroni level). Alignment peaks at layer 20 (74% depth), suggesting visual grounding is encoded in mid-to-late layers before final decoding reshapes representations into text.

Table 5: Per-token-type attention entropy (SFT vs. DPO R3). Texture tokens show the largest reduction.

Token Type	SFT Entropy	DPO R3 Entropy	Δ
State tokens	7.151	7.137	-0.014
Texture tokens	7.091	7.047	$\mathbf{-0.044}$
Other tokens	6.970	6.944	-0.026

Table 6: Cross-modal alignment between CLIP visual space and description edit-distance space. Selectivity = non-neighbor ED / neighbor ED.

Round	Spearman $\rho \uparrow$	p-value	Neighbor ED $\downarrow$	Non-Neighbor ED	Selectivity $\uparrow$
SFT (R0)	0.116	< 0.000001	2.155	2.638	1.224
DPO R1	0.121	< 0.000001	1.930	2.414	1.251
DPO R2	0.146	< 0.000001	1.711	2.220	1.297
DPO R3	**0.179**	< 0.000001	1.858	2.383	1.283

A.3 Extra Domain Examples

Figure 6 shows a representative input from each of the three additional visual domains. The pipeline is identical to the time-series setting.

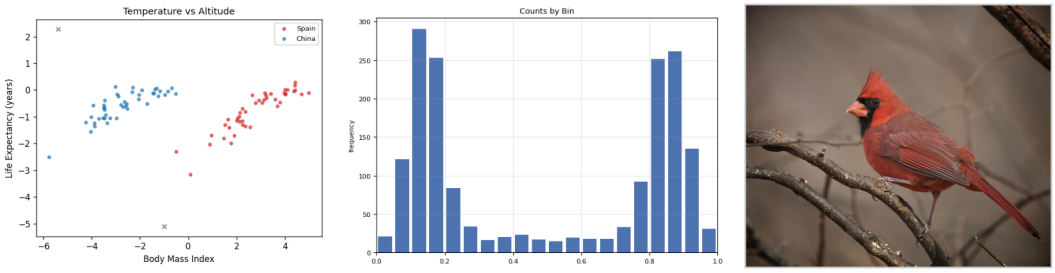


Figure 6: Example inputs from the additional visual domains. **Left:** scatterplot with two positively trending clusters and a small number of isolated outliers. Titles, axis labels, and legend entries are randomized and carry no information about the plotted data, so the model must describe the points themselves rather than the surrounding text. **Middle:** bimodal histogram with two narrow modes separated by a wide, low valley. **Right:** CUB-200-2011 photograph (Northern Cardinal). Neighborhoods for this domain are built from DINOv2 image embeddings, with no underlying signal.

A.3.1 Neighborhood Size Sensitivity

We vary the number of structural neighbors K while holding the remaining pipeline fixed. All tested values improve over SFT on both financial tasks, with the strongest gains at $K = 10$. Smaller neighborhoods provide noisier candidate consensus, while larger ones admit less similar examples and dilute the structural signal; crash-risk detection is more sensitive to K than large-move detection.

Table 7: Spearman ρ between CLIP cosine similarity and LLM hidden-state cosine similarity at each layer. Depth % = layer / 27. All correlations $p < 10^{-14}$.

Checkpoint	Layer 6 (22%)	Layer 13 (48%)	Layer 20 (74%)	Layer 27 (100%)
SFT (R0)	0.059^{***}	0.151^{***}	$\mathbf{0.245}^{***}$	0.200^{***}
DPO R3	0.116 (+97%)	0.184 (+22%)	**0.256** (+4%)	0.232 (+16%)

Table 8: Sensitivity to neighborhood size K. Relative improvement in accuracy over SFT (%) on the financial downstream tasks (Qwen2-VL-7B, seed 42). $K = 10$ is the setting used throughout the paper.

K	Crash risk	Large move
5	+7.1	+28.6
10	**+14.3**	**+31.0**
20	+3.6	+21.4

A.4 LLM USAGE

Claude Code (Anthropic Sonnet/Haiku/Opus) assisted in drafting some code throughout this research, producing code for files that was then reviewed by a human reader. Also helped for polishing and grammar checking parts of the document.

A.5 ADDITIONAL VISUAL DOMAINS

We additionally evaluate on histograms, scatterplots, and CUB bird images.

Table 9: **Zero-shot transfer to additional visual domains.**

Domain	Held-out evaluation	Relative gain over SFT
Histograms	Distribution trend recognition	+35.2%
Scatterplots	Regional structure and anomaly recognition	+20.0%
CUB birds	Fine-grained attribute verification	+18.8%

For histograms and scatterplots, neighborhoods use the underlying data representation; for CUB, neighborhoods use DINOv2 image embeddings. Prompts and vocabularies are generated from unlabeled examples and fixed before training.

A.6 ABLATION FULL RESULTS

A.6.1 PROMPT AND OUTPUT FORMAT ROBUSTNESS

We test sensitivity to prompt wording and output representation across domains.

Table 10: **Sensitivity to prompt wording and output format.** Relative improvement over SFT (%) on held-out downstream tasks. Finance reports large-move / crash-risk detection.

Prompt / output	**Finance** **Large move / crash risk**	**Scatterplots**	**Histograms**	**CUB**
Curated	31.0 / 14.3	20.0	35.2	18.8
General	30.4 / 13.4	15.7	26.5	17.6
LLM-generated, least detailed	30.6 / 13.4	17.0	27.2	15.7
LLM-generated, semi-detailed	30.6 / 14.3	11.7	31.1	16.3
LLM-generated, very detailed	30.1 / 13.4	13.0	27.1	15.4
Free text	28.3 / 10.7	11.2	29.2	16.7

We additionally compare structured tuples with a simpler single-state representation on the financial tasks while holding the neighbor graph and training pipeline fixed.

Table 11: **Output representation ablation on financial downstream tasks.** Structured tuple and single-state variants use the same neighbor graph and preference-learning pipeline.

Format	Description	Crash Risk		Large Move	
		Acc	F1	Acc	F1
SFT (R0)	—	0.56	0.49	0.42	0.36
Structured tuple	`(state, texture)` pairs	**0.64**	**0.64**	**0.55**	**0.52**
Single state	Combined token (e.g. `rise_gradual`)	0.64	0.64	0.48	0.47

We also reduce the financial vocabulary from 7 states and 4 textures to 4 states and 2 textures. Performance remains above SFT, with relative gains of 27.4% on large-move detection and 10.4% on crash-risk prediction. These results demonstrate that the preference signal does not depend on a particular prompt, output format, or vocabulary size.

A.6.2 Neighborhood Size Sensitivity

We test sensitivity to the number of structural neighbors K while holding the remaining pipeline fixed.

Table 12: **Sensitivity to neighborhood size K.** Relative improvement over SFT (%) on financial downstream tasks.

K	Crash Risk	Large Move
5	7.1	28.6
10	**14.3**	**31.0**
20	3.6	21.4

Performance improves over SFT for all tested values of K, with the strongest gains at $K = 10$.

A.7 Dataset and Evaluation Details

The financial domain contains 4,036 price charts rendered from CRSP time-series data. The ECG domain contains 3,997 PTB-XL charts spanning five diagnostic superclasses, with an equal overall normal/abnormal ratio. The seismic domain contains 4,500 traces, split evenly between earthquake and noise examples.

A.7.1 Normalized vs. Raw Edit Distance

The scoring function uses normalized edit distance (NED $= \mathrm{ED}/\max(|s_1|, |s_2|)$) rather than raw ED to ensure scores reflect structural dissimilarity rather than description length. To quantify the bias introduced by raw ED, we re-score held-out preference pairs at each of three DPO rounds using both metrics and compare the resulting preference labels and margin distributions.

Table 13: NED vs. raw ED scoring bias on CRSP preference pairs across DPO rounds. Pair flip rate = fraction of pairs where NED and raw ED assign different chosen/rejected labels. Length-margin Spearman ρ measures correlation between length difference and preference margin.

Round	Pair Flip Rate		Length-Margin ρ		Mean Margin
	NED	Raw ED	NED	Raw ED	NED / Raw
R1	16.6%		-0.001	$+0.050$	0.208 / 0.756
R2	16.1%		-0.078	$+0.055$	0.208 / 0.762
R3	17.1%		-0.042	-0.092	0.215 / 0.746

Raw ED consistently inflates preference margins 3.5× over NED (0.21 vs. 0.75), driven by length rather than structural distance: the length-margin Spearman correlation is near zero under NED but

positive under raw ED at R1–R2. Across rounds, 16–17% of pairs receive different chosen/rejected labels under the two metrics, meaning raw ED systematically corrupts roughly one in six preference pairs. When the DPO model is trained on raw-ED pairs, crash-risk prediction collapses to near the SFT baseline, consistent with the model learning to generate longer descriptions rather than structurally accurate ones.

A.7.2 Non-Structural and Degenerate Training Variants

We evaluated several training variants where the preference signal lacks a structural visual grounding component. Each variant degrades in a characteristic way, confirming that the structural-neighbor signal is the load-bearing component of the pipeline.

No-image preference (text-only judge). In this variant, the scoring model judges candidate descriptions by text quality alone, with no access to the chart image. Preference pairs are generated and DPO training proceeds identically to the full pipeline. Table 14 reports clustering metrics across three rounds.

Table 14: Clustering metrics under text-only (no-image) preference scoring. The judge has no access to the chart image; preferences reflect text fluency only. SFT baseline sep. ratio = 3.60.

Round	Sep. Ratio	Silhouette	Families
R1	1.02	-0.039	20
R2	1.16	-0.121	20
R3	1.02	-0.175	15

Separation ratio falls far below the SFT baseline of 3.60 in all three rounds, and silhouette deteriorates monotonically to -0.175 at R3. Without the chart image, DPO pushes toward text fluency while moving away from visual grounding: the model learns to produce grammatically clean descriptions that are uncorrelated with the underlying chart structure. Family count shrinks from 20 to 15, indicating progressive mode collapse rather than convergence.

Random preference. Assigning chosen/rejected labels uniformly at random removes all signal from the preference pairs while keeping the training procedure otherwise identical. After three rounds, sep. ratio reaches 0.81 (far below SFT at 3.60) and silhouette is -0.040, indicating that random preference training actively degrades the clustering structure inherited from SFT fine-tuning.

Iterative SFT (no contrastive preference). Standard iterative SFT, without the DPO contrastive objective, collapses the output format across all trials: the model converges on degenerate looping sequences (e.g., `[([([([([ ...])` at inference time. Without a contrastive preference signal to distinguish between structurally different descriptions, the model learns to exploit format repetition as a low-loss solution.

SFT with LoRA across all layers (including vision encoder). Applying LoRA to all transformer layers, including the vision encoder, produces zero valid tuple outputs from 800 generation attempts, with training loss failing to converge. The pretrained visual representations that the structural-neighbor preference signal is designed to reorganize are corrupted by the unconstrained weight updates, preventing the model from learning any consistent output format.

Downstream task results. Table 15 consolidates crash-risk and large-move detection performance across all non-structural variants alongside the full pipeline and base model. Variants that lack a visual structural signal produce results indistinguishable from the base model, with all differences non-significant ($p > 0.3$, bootstrap CI). Iterative SFT and LoRA-on-encoder variants are excluded as they produce no valid outputs.

The gap between the full pipeline (crash-risk F1 = 0.64, large-move F1 = 0.52) and all non-structural variants ($\leq$0.50 and $\leq$0.37 respectively) confirms that downstream gains are attributable specifically to visual structural grounding, not to the DPO training procedure itself or to text-quality improvements.

Table 15: Downstream task performance for non-structural and degenerate variants vs. full pipeline (DPO R3) and base model. All variants evaluated at their final round (R3). † = $p > 0.3$ vs. base (bootstrap); no statistically significant difference from base.

Variant	Signal	Crash Risk		Large Move	
		Acc	F1	Acc	F1
SFT (R0)	—	0.56	0.49	0.42	0.36
No-image pref. (R3)	Text fluency only	$0.57^{\dagger}$	$0.50^{\dagger}$	$0.43^{\dagger}$	$0.37^{\dagger}$
Random pref. (R3)	None (uniform random)	$0.55^{\dagger}$	$0.48^{\dagger}$	$0.41^{\dagger}$	$0.35^{\dagger}$
Full pipeline (DPO R3)	Structural neighbor	**0.64**	**0.64**	**0.55**	**0.52**

A.7.3 Iterative vs. Matched-Compute Single-Round DPO

To isolate the effect of iterative re-description from simply training longer, we train a single-round DPO baseline using the same total compute: 9 epochs on SFT-generated preference pairs (680 pairs $\times$ 9 epochs / 4 grad_accum = 1,530 optimizer steps, identical to 3 rounds $\times$ 510 steps).

Training dynamics. Table 16 shows that single-round DPO overfits the fixed preference pairs: validation loss bottoms at epoch 6 then rises, and the train/val gap widens monotonically from $+0.008$ to $+0.083$.

Table 16: Single-round DPO per-epoch training dynamics. Val loss rises after epoch 6; train/val gap widens monotonically, indicating memorization of fixed SFT-generated pairs.

Epoch	Train Loss	Val Loss	Train/Val Gap	Reward Acc
1	0.688	0.676	-0.012	54.9%
3 *(= 1 iter. round)*	0.537	0.545	$+0.008$	75.4%
6 *(= 2 iter. rounds)*	0.458	0.503	$+0.045$	79.4%
9 *(= 3 iter. rounds)*	0.428	0.511	$+0.083$ ↑	81.3%

Iterative DPO avoids this: each round receives fresh preference pairs from the updated model, resetting the learning curve. Table 17 shows the per-round dynamics; the train/val gap at epoch 3 stays small (-0.004–$+0.015$) across all rounds.

Table 17: Iterative DPO per-round training dynamics. Each round resets to ≈0.68 train loss at epoch 1 because the model receives genuinely harder, fresher preference pairs.

Round	Epoch	Train Loss	Val Loss	Train/Val Gap	Reward Acc
Iter R1	1	0.679	0.653	-0.026	59.7%
	2	0.610	0.599	-0.011	72.4%
	3	0.569	0.584	$+0.015$	74.9%
Iter R2	1	0.674	0.632	-0.042	61.7%
	2	0.587	0.553	-0.034	75.0%
	3	0.536	0.532	-0.004	77.5%
Iter R3	1	0.680	0.649	-0.031	59.9%
	2	0.592	0.559	-0.033	74.1%
	3	0.539	0.536	-0.003	77.0%

Preference pair quality. Because iterative DPO re-describes training charts each round with the updated model, preference pairs become progressively harder. Table 18 shows that the fraction of high-margin pairs (> 0.3) grows from 15.1% to 19.0% across rounds, and the description vocabulary shrinks as the model converges. Single-round trains on the same SFT-level pairs for all 9 epochs and never benefits from this curriculum.

Table 18: Preference pair statistics per round. Iterative DPO generates increasingly discriminative pairs; single-round replays the same low-margin pairs for all 9 epochs.

Round	Description Source	Mean Margin ↑	Hard Pairs (> 0.3) ↑	Unique Vocab ↓
Iter R1	SFT model	0.218	15.1%	334
Iter R2	R1 model	0.223	16.5%	251
Iter R3	R2 model	**0.227**	**19.0%**	**203**
Single (9 ep)	SFT model (frozen)	0.218	15.1%	334

Downstream task performance. Table 19 reports crash-risk and large-move detection for the single-round model alongside the full iterative pipeline and base model. Single-round underperforms iterative R3 on crash-risk detection (-0.064 in both accuracy and F1), while the two are essentially tied on large-move detection (0.528 vs. 0.55 accuracy, 0.521 vs. 0.52 F1). The crash-risk gap is consistent with the overfitting pattern in training dynamics: high reward accuracy (81.3%) on memorized pairs does not transfer to held-out detection.

Table 19: Downstream task performance: base model, single-round DPO (9 ep, matched compute), and full iterative DPO R3. Single-round results evaluated on the same held-out split.

Checkpoint	Crash Risk		Large Move	
	Acc	F1	Acc	F1
SFT (R0)	0.56	0.49	0.42	0.36
Single-round (9 ep)	0.576	0.576	0.528	0.521
Iterative R3	**0.64**	**0.64**	**0.55**	**0.52**

Cross-modal alignment. Table 20 shows that iterative R3 achieves Spearman $\rho = 0.179$, outperforming single-round (9 epochs, identical compute) at $\rho = 0.154$—equivalent to iterative R2's level despite the same total gradient steps. The curriculum of re-description and re-scoring extracts more alignment signal per step than repeated replay.

Table 20: Cross-modal alignment: iterative DPO vs. matched-compute single-round. $n = 20{,}000$ pairs (Spearman); computed on training-set charts with fixed CLIP visual graph.

Checkpoint	Spearman ρ ↑	p-value	Neighbor ED ↓	Selectivity ↑
SFT (R0)	0.116	$p < 0.00001$	2.155	1.224
Iter R1	0.121	$p < 0.000001$	1.930	1.251
Iter R2	0.146	$p < 0.000001$	1.711	1.297
Iter R3	**0.179**	$p < 0.000001$	1.858	1.283
Single (9 ep)	0.154	$p < 0.000001$	1.712	1.324

A.8 Maintains General VLM Reasoning Capabilities

Each DPO round updates the reference model to the previous round's checkpoint rather than anchoring to the original SFT model throughout training. A natural concern is whether this progressive updating degrades general vision-language capabilities outside the targeted time-series tasks. To check this, we evaluate MMLU (5-shot) on the base model, SFT checkpoint, and each DPO round. Scores remain flat across all rounds, with no statistically significant change ($p > 0.4$, paired bootstrap). The LoRA rank-16 updates targeting only attention projection layers, combined with the small per-round step count (≈510 optimizer steps), appear sufficient to prevent drift in general language understanding while allowing the structural visual grounding signal to accumulate across rounds.

A.9 ECG Domain: Downstream Analysis Details

A.9.1 Morphology QA Benchmark

We evaluate on 1,555 programmatic multiple-choice questions across 14 morphology types, with gold labels derived directly from the raw signal (no model involvement). Table 21 reports per-type accuracy for the base model and DPO R3.

Table 21: ECG morphology QA benchmark: per-question-type accuracy (base vs. DPO R3). Gold labels computed from raw PTB-XL signal. "Class-collapse" = both models always predict the majority class.

Question Type	N	Base	DPO R3	Δ
Genuine DPO gains				
T-wave polarity	86	44.2%	**73.3%**	+29.1pp
Waveform closing trend	105	36.2%	**48.6%**	+12.4pp
Dominant feature position	129	86.0%	**90.7%**	+4.7pp
QRS width (spike vs. peak)	131	86.3%	87.0%	+0.7pp (minority recall 27% → 41%)
ST-segment shape	88	60.2%	55.7%	−4.5pp acc (sloping recall 0% → 46%)
Stable / class-collapse in both models				
QRS polarity	129	93.8%	89.1%	−4.7pp (near ceiling)
Post-QRS direction	87	44.8%	44.8%	0 (both predict "downward")
Pre-QRS feature	119	78.2%	77.3%	−0.9pp (both predict "bump")
Q-wave, S-wave, P-wave, etc.	—	No meaningful change; class-collapse or yes-bias in both		
Overall	1555	60.8%	**62.7%**	+1.9pp

DPO gains are concentrated in questions requiring the model to identify a specific visual feature at a known position in the waveform sequence—exactly the information encoded by the tuple descriptions. T-wave polarity and closing trend show the largest gains (+29.1pp and +12.4pp respectively). ST-segment accuracy decreases by 4.5pp in aggregate but the model gains genuine discrimination: sloping recall improves from 0% to 46%, while the base model always predicts the majority class (flat).

A.9.2 Clinical Alignment and Abnormality Screening

Table 22 reports cluster purity and NMI against PTB-XL diagnostic superclasses. The model is clustered on description similarity; diagnostic labels are used only as a post-hoc alignment check.

Table 22: ECG clinical alignment: description cluster purity and NMI vs. PTB-XL diagnostic superclass (NORM/CD/STTC/HYP/MI). Random baseline purity ≈ 0.23 for 5 classes. Agglomerative result forces 0% noise by assigning all 134 held-out ECGs.

Checkpoint	Clusters	Noise Rate	Cluster Purity ↑	NMI ↑
Random baseline	—	—	0.231	≈0.000
DPO R1	8	46%	0.381	0.097
DPO R2	7	39%	0.388	0.101
DPO R3	10	43%	**0.396**	**0.149**
DPO R3 (agglomerative)	26	0%	**0.493**	**0.216**