DATA-WEIGHT DIRECTIONS ROTATE ALONG TRAINING: HORIZON-MATCHED ATTRIBUTION FOR GUI TRAJECTORIES

Anonymous authors
Paper under double-blind review

ABSTRACT

When attribution scores determine training-data weights, their effects depend on how long training continues. We ask whether weights selected after K updates remain beneficial after $T > K$ updates. We formulate attribution as a derivative through a finite training run. Our method, horizon-matched attribution, evaluates candidate weights at the intended training endpoint. We compare short- and terminal-horizon choices under the same training recipe, using paired weight changes that keep the total weight unchanged. Across six Qwen3-VL Mind2Web training subsets, cosine similarity between the short- and terminal-horizon directions is always negative. Weights selected at 88 updates lower loss on the dataset used for selection on all six subsets but raise it at 701 updates. Terminal selection lowers held-out offline expert-action cross entropy on the task, website, and domain panels for every subset. On three subsets, the failure persists across several early selection horizons, while later selection recovers positive mean terminal gains. Fixed choices reproduce the reversal under two additional optimizer seeds on those subsets. InternVL3.5-2B also favors terminal selection, while controlled Android experiments separate sign correction from combining directions. Matching the measurement horizon to the training endpoint corrects short-horizon weighting failures in these GUI fine-tuning settings. Code and results are available at `https://anonymous.4open.science/r/horizon-matched-attribution-3C50/`.

1 INTRODUCTION

GUI agents are fine-tuned on action trajectories from different tasks, websites, and applications (Deng et al., 2023; Li et al., 2024). Data attribution can decide which trajectories receive more weight. The relevant test is whether those weights improve action prediction after the planned training run, not just loss at an early checkpoint. A short run can compare candidates more cheaply, but an early winner may raise final loss. Does a weighting that helps after K updates still help after $T > K$ updates?

Reweighting changes the parameters, optimizer moments, and gradients that govern later updates. Prior work establishes that data influence changes along training (Wang et al., 2025a; Lee et al., 2026). Short-horizon objectives can also bias hyperparameter and data-weight optimization (Wu et al., 2018; Das et al., 2026). Attribution studies connect selection performance to optimizer fidelity and look-ahead length (Deng et al., 2026). This study examines the transfer of a fixed weighting under the same task and training recipe. The comparison separates the accuracy of short-run evaluation from the value of the resulting choice for longer training.

We propose *horizon-matched attribution*: evaluate candidate data weights at the intended training endpoint. Let $\theta_H(\alpha)$ denote the parameters after H updates with trajectory weights $w(\alpha)$ and a fixed training recipe. A separate dataset Q, called the selection query, is used to select weights. The direction that locally reduces its loss is

$$d_H = -\nabla_\alpha \mathcal{L}_Q(\theta_H(\alpha))\,\big|_{\alpha=0}, \tag{1}$$

where $\alpha = 0$ gives the base weights. Changing H can rotate d_H, not just rescale it. To first order, terminal gain depends on the alignment of short and terminal directions within the measured subspace.

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

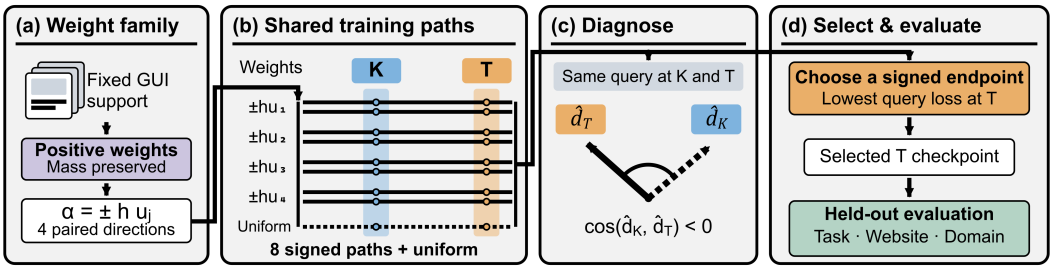

Figure 1: **Overview of horizon-matched attribution.** (a) A fixed training subset defines positive weights along four paired directions. (b) Eight signed paths and a uniform reference share the training recipe, with checkpoints at K and T. (c) Directions estimated on the same query can disagree across horizons. (d) The terminal rule selects the measured weighting with lowest query loss at T, then evaluates it on held-out data. The short rule applies the same selection criterion at K.

Negative alignment predicts a local increase in terminal loss. We then test whether the selected weights reverse their effect when training continues to T.

The experiments focus on supervised GUI fine-tuning. Each training example contains a screenshot, task instruction, action history, and reference next action. For each training subset, we construct a low-dimensional family of positive weights whose total remains fixed. Paired training runs measure the effects of weight changes at a short horizon K and a terminal horizon T. The two rules choose among these candidates using loss at K or T, then compare their selected weights after the same T updates. Both rules use the same training trajectories, set of candidate weights, training order, target-token budget, random settings, and selection query. Held-out losses are used only for evaluation. Figure 1 illustrates this comparison.

On Multimodal-Mind2Web (Deng et al., 2023), cosine similarity between the short- and terminal-horizon directions is negative on all six training subsets. Selection at $K = 88$ lowers selection-query loss on 6/6 subsets but raises it at $T = 701$ on 6/6. Selection at T lowers both selection-query loss and held-out expert-action cross entropy on the task, website, and domain panels for every subset. Keeping the choices fixed reproduces the reversal and terminal advantage under two new optimizer seeds on the first three subsets. A six-horizon sweep on those subsets finds negative mean terminal held-out gains for selection at 44, 88, 176, and 350 updates. The means turn positive at 525 and 701.

InternVL3.5-2B (Wang et al., 2025b) also favors terminal selection under the same Mind2Web data roles and horizons. AndroidControl studies (Li et al., 2024) compare sign correction with combining directions and evaluate fresh trajectories from seen and unseen applications. A schedule control shows that equal cumulative weights can produce different Adam training outcomes.

The paper makes three contributions:

- We make the training endpoint explicit in data-weight attribution. Our derivation links cross-horizon direction alignment to the local change in terminal loss, explaining how a weight change can help early yet hurt later.
- We show that weights selected from accurately measured short-run gains can increase loss after longer GUI fine-tuning. This separates accurate early evaluation from effective long-run data selection: better estimates of early gains need not resolve the mismatch.
- We test horizon-matched selection, which restores held-out expert-action loss gains on Mind2Web across two multimodal models. These results motivate evaluating data-selection methods at the intended training endpoint and provide a measured reference for assessing cheaper approximations.

2 Related Work

Attribution along training. Influence functions approximate leave-one-out effects near a fitted solution (Koh & Liang, 2017); TracIn accumulates gradient agreement across checkpoints

(Pruthi et al., 2020). The attribution target also depends on the counterfactual being approximated (Bae et al., 2022). Unrolling and metagradient methods account for training dynamics when estimating influence or optimizing data contributions (Chen et al., 2021; Bae et al., 2024; Ilyas & Engstrom, 2025; Engstrom et al., 2025). DVEmb captures the influence of removing a sample at a particular training iteration (Wang et al., 2025a), while stagewise attribution studies non-monotonic influence and sign changes (Lee et al., 2026). AdamW analyses further show how optimizer memory changes the timing and sign of a minibatch's effects (Liu & Li, 2026; Guo, 2026). These results establish why attribution depends on the training path; our experiments examine the consequences for weights selected using that attribution.

Short-horizon bias and data weighting. Wu et al. (2018) show that short meta-optimization horizons can favor learning rates with poor long-run performance. Truncated bilevel differentiation studies when approximate hypergradients support optimization (Shaban et al., 2019). More directly, Das et al. (2026) analyze repeated domain-weight updates: one-step inner runs can fail, while longer runs improve convergence bounds under strongly convex per-domain losses. Deng et al. (2026) connect data-selection performance to optimizer mismatch, attribution approximation error, and look-ahead length. The comparison in this study holds each candidate weighting fixed throughout training and scores its actual losses at K and T. This distinguishes failure of a short-run weighting choice from errors in estimating its short-run effect.

Data selection and later performance. LESS uses optimizer-aware gradient similarity for targeted instruction tuning (Xia et al., 2024); influential-subset studies examine the reliability of such selections (Hu et al., 2024). Dynamic bilevel methods update data weights as training proceeds (Yu et al., 2025). CUPID curates robot demonstrations using influence functions (Agia et al.). GUI-CIDER filters synthesized GUI knowledge by causal saliency and semantic density before mid-training (Wu et al., 2026). Yang et al. (2026) study how selection for a current task affects later adaptation and forgetting across a sequence of tasks. Our horizon comparison keeps the training task, selection query, and occurrence schedule fixed. It tests whether the same persistent weights change from beneficial to adverse as optimization continues, without requiring a change in task.

Measuring whether weights transfer. Our paired measurements build on orthogonal perturbations and surrogate-guided search (Choromanski et al., 2018; Maheswaranathan et al., 2019; Meier et al., 2019). The same runs estimate direction changes and provide candidate losses for selection. We link projected direction disagreement to measured finite-weight reversal, then compare short and terminal selection on held-out expert-action loss at a common training endpoint.

3 METHOD

Given a training subset (the support), a selection query, and horizons $K < T$, we construct candidate weights and train each candidate to T. Query loss at K selects the short-horizon weights; query loss at T selects the terminal weights. Features, measurement directions, and selected weights are fixed before held-out evaluation at T. Paired candidates also estimate how the direction changes with horizon. This comparison provides a diagnostic; selection uses the measured losses directly.

3.1 DECISION SETTING AND FEASIBLE WEIGHT FAMILY

We keep the training trajectories fixed to separate reweighting effects from changes in sample coverage. Let the support contain trajectories $i = 1, \dots, n$, with $G(i)$ denoting trajectory i's group in $\mathcal{G}$. A group is an application in the Android studies; Mind2Web uses one global group. Each trajectory has a base weight $q_i > 0$, with total group weight $Q_g = \sum_{i \in g} q_i$. In all experiments, $Q_g = 1/|\mathcal{G}|$ and the base weights are uniform within a group. Fixed feature coordinates $z_i \in \mathbb{R}^d$ parameterize the positive exponential tilt

$$w_i(\alpha) = Q_{G(i)} \frac{q_i \exp(z_i^\top \alpha)}{\sum_{j \in G(i)} q_j \exp(z_j^\top \alpha)}, \qquad \alpha \in \mathbb{R}^d. \tag{2}$$

The vector α parameterizes trajectory weights through their features, rather than model parameters. The normalization gives $w_i(0) = q_i$ and $\sum_{i \in g} w_i(\alpha) = Q_g$ exactly. Positive weights retain every

trajectory, while fixed group masses prevent a change in overall objective scale or group balance. We divide each trajectory's mass across its action steps and scheduled occurrences. Its normalized share remains $w_i(\alpha)$; backpropagation applies a common factor T, as specified in Appendix A.

We center trajectory features within groups, remove redundant coordinates, and whiten the retained coordinates. To limit training costs, the main experiments use $m = 4$ orthonormal directions $u_1, \ldots, u_m$. The first follows a checkpoint score based on whether a trajectory's AdamW update reduces query loss locally. The remaining directions are sampled with a fixed seed and orthogonalized. Appendix A gives the features and numerical construction; Appendix B.1.1 extends the measurement to six directions.

Let $\tau = (M, \theta_0, \mathrm{Opt}, \sigma, \xi)$ collect the model, initial parameters, optimizer with its initial state, occurrence schedule including batch boundaries, and stochastic choices. Training for H updates gives

$$\theta_H(\alpha; \tau) = \mathcal{A}_H(\tau; w(\alpha)), \qquad F_H(\alpha) = \mathcal{L}_Q(\theta_H(\alpha; \tau)), \qquad d_H = -\nabla_\alpha F_H(0), \tag{3}$$

where $\mathcal{L}_Q$ is macro-averaged action-token cross entropy on a selection query Q. The derivative accounts for how changing data weights affects all H updates and the resulting query loss. We estimate directional derivatives from paired training runs below, without backpropagating through the full training computation. The direction d_H depends on the weight family and training recipe τ.

3.2 First-order horizon diagnostic

Assuming F_T is differentiable at $\alpha = 0$, for any unit direction v and sufficiently small $\epsilon > 0$, a first-order expansion at the base weights gives

$$\mathcal{G}_T(v) \equiv F_T(0) - F_T(\epsilon v) = \epsilon d_T^\top v + o(\epsilon). \tag{4}$$

We measure the subspace spanned by the u_j. Its orthogonal projector is $P = \sum_{j=1}^m u_j u_j^\top$, and the projected descent direction is $d_H^U = P d_H$. For v in this subspace, $d_T^\top v = (d_T^U)^\top v$. When d_K^U and d_T^U are nonzero,

$$\mathcal{G}_T\left(\frac{d_K^U}{\|d_K^U\|}\right) = \epsilon \|d_T^U\| \cos(d_T^U, d_K^U) + o(\epsilon). \tag{5}$$

A negative cosine predicts a terminal loss increase for a sufficiently small step along the projected short direction. The matched direction $d_T^U / \|d_T^U\|$ instead reduces loss locally when $d_T^U \neq 0$. We separately test whether the selected finite weight changes reverse their effect.

3.3 Paired path measurements

For measurement radius h, we train the paired candidate weightings $\alpha = \pm h u_j$ once to T and retain checkpoints at each measurement horizon H, including K and T. The radius is measured in the whitened feature coordinates, not as a percentage change in trajectory weights. The positive and negative paths provide a central difference along each direction. Their selection-query losses give

$$\widehat{g}_{H,j} = \frac{F_H(+h u_j) - F_H(-h u_j)}{2h}, \qquad \widehat{d}_H = -\sum_{j=1}^m \widehat{g}_{H,j} u_j. \tag{6}$$

The finite-difference estimate $\widehat{d}_H$ approaches d_H^U as $h \to 0$ for differentiable F_H. When both estimates are nonzero, we report their cosine similarity at the measured radius:

$$c(K, T) = \frac{\widehat{d}_K^\top \widehat{d}_T}{\|\widehat{d}_K\| \, \|\widehat{d}_T\|}. \tag{7}$$

Within a support, the uniform and signed paths share τ; only $w(\alpha)$ changes. The cosine therefore compares horizons within the same weight space.

3.4 Turning measurements into data weights

Our primary rule selects the candidate with the lowest measured query loss. Define

$$\mathcal{C}_h = \{+h u_1, -h u_1, \ldots, +h u_m, -h u_m\}, \qquad \alpha_H^{\mathrm{dir}} \in \arg\min_{\alpha \in \mathcal{C}_h} F_H(\alpha). \tag{8}$$

The short and terminal rules use $H = K$ and $H = T$, respectively. The same rule applies at intermediate horizons. Uniform weighting ($\alpha = 0$) stays outside the candidate set as a fixed comparator. The selected adapter already exists among the measured paths under the selection seed. To test seed transfer, we keep α_H^{dir} fixed and retrain with new seeds and the same schedule.

On the selection query, $F_T(\alpha_T^{\mathrm{dir}}) \le F_T(\alpha_H^{\mathrm{dir}})$ follows from the argmin. Experiments test whether the selected weights improve over uniform and whether the benefit persists on held-out data or under new seeds.

The secondary projection rule tests whether combining measured coordinates improves over choosing one. For a prescribed radius ρ and nonzero $\widehat{d}_H$, it uses

$$\alpha_H^{\mathrm{proj}} = \rho \frac{\widehat{d}_H}{\|\widehat{d}_H\|}, \tag{9}$$

and requires one additional training run for each selected vector. Equation (5) motivates this direction.

4 Experimental Design

4.1 Questions and controls

We examine how weight directions change with training horizon, whether short-selected weights remain beneficial at T, and whether terminal selection improves held-out loss. Uniform weighting is the common comparator. Within each paired comparison, paths share the training initialization, occurrence stream, update count, and processed target-token budget.

Our metric is offline expert-action cross entropy. We first average over each trajectory's action tokens, then over trajectories within each website or application, and finally over websites or applications. We report uniform-minus-method loss, so positive gains favor reweighting. Descriptive intervals summarize variation across support draws. Appendix A gives the full configuration and aggregation rules.

4.2 Mind2Web study

On Multimodal-Mind2Web, we test whether weights selected early remain beneficial after longer Qwen3-VL-2B training with rank-2 LoRA. Each support draws 25 trajectories from a pool of 250 training trajectories. A disjoint set of 60 trajectories supplies the selection query, and the official `test_task`, `test_website`, and `test_domain` splits contribute 100 evaluation trajectories each. These panels test transfer to new tasks, websites, and domains. Inputs contain a screenshot, task instruction, and up to four previous reference actions; the target is the demonstrated next-action string.

For each support, four paired directions with $h = .1$ define eight candidate weightings. These paths and the uniform comparator each process 47,004 supervised target tokens in 701 updates. We measure candidates at $K = 88$ and $T = 701$ and compare the selected weights at the common endpoint T. We report six supports: three initial draws and three additional draws whose seeds were fixed before their model outcomes were observed. The supports can overlap and share query and evaluation panels, so they test robustness to the training subset.

To test whether the result depends on one particular short horizon, we measure $H \in \{44, 88, 176, 350, 525, 701\}$ on the first three supports. The candidate weights, training paths, and evaluation panels stay fixed. At each horizon, query loss selects a candidate, and we evaluate its terminal adapter. The horizon grid was specified before the sweep; this extension uses the existing supports.

To assess sensitivity to training randomness, we retrain the uniform, short-selected, and terminal-selected conditions under two additional optimizer seeds on those same three supports. We keep the weights fixed without reselection. The source seed is reported separately, and transfer summaries average the two new seeds within each support.

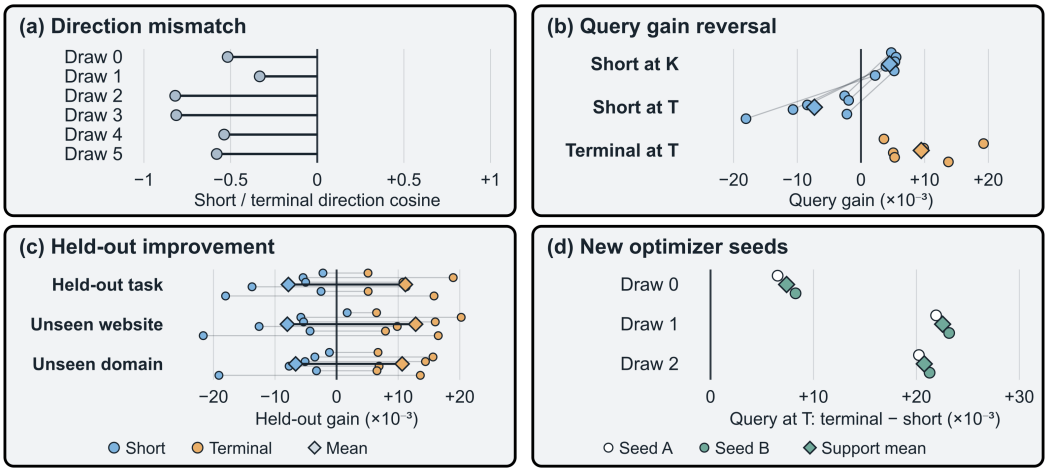

Figure 2: **Qwen3-VL-2B weighting effects on Mind2Web.** (a–c) Six supports under the selection seed: (a) short/terminal direction cosines, (b) query effects at $K = 88$ and $T = 701$, and (c) held-out effects at T. In (b–c), gains are uniform-minus-rule expert-action cross entropy ($\times 10^{-3}$; higher is better); blue/orange indicate short/terminal selection. Circles mark supports and diamonds their means; thin lines pair supports and bold lines in (c) join means. (d) Terminal-over-short query gain at T after retraining fixed choices on the first three supports with two new optimizer seeds: A (open circles) and B (filled circles). Diamonds average seeds within each support.

4.3 Model replication and Android controls

We test whether the Mind2Web findings extend to another multimodal model using InternVL3.5-2B on the first three supports. The data roles, occurrence streams, training horizons, and candidate-set size are unchanged. Scores, feature coordinates, and candidate directions are recomputed for InternVL, which also uses rank-2 LoRA.

The AndroidControl studies examine what correcting the weighting requires. Using Qwen2.5-VL-3B with 500 training and 100 query trajectories, we compare uniform weighting, a checkpoint-local direction, its paired sign correction, and projections combining four directions measured at K or T. Official in-distribution and app-unseen panels evaluate whether changing the sign is sufficient or whether combining directions yields further gains.

After short calibration fails on three Android supports, a follow-up measures terminal candidates on a fresh query bank and evaluates them on fresh same-application and app-unseen validation trajectories. It retains the supports and training schedules, but changes the query and validation banks as well as the measurement horizon. Single-axis and multi-direction choices test whether terminal calibration also needs information from more than one direction. A separate schedule control compares identical cumulative weights under different occurrence schedules, motivating the matched schedules in the main experiments. Appendix A gives the Android protocols, and Appendix D details the schedule comparison.

5 Results

5.1 Terminal selection restores gains after short-horizon reversal

Weights selected at K reverse their effect by T (Figure 2b). On all six supports, they lower selection-query loss at K (mean gain $+.0045$) but raise it at T (mean $-.0073$). Weights selected at T lower terminal query loss on all six supports (mean $+.0095$).

Terminal selection also improves held-out loss at T (Figure 2c; Table 1). Short selection has negative gains on held-out tasks and unseen domains for all six supports, and on unseen websites for five of six. Terminal selection is positive for every support in every panel, with mean gains between $+.0106$ and

Table 1: Qwen3-VL-2B on six Mind2Web supports. Values are uniform-minus-method website-macro action-token cross entropy; positive is better. Parentheses show the number of positive support effects out of six; descriptive intervals are reported in Appendix B.1. Differences are computed before rounding.

Method	Held-out task	Unseen website	Unseen domain
Short selection	$-.0078$ (0/6)	$-.0080$ (1/6)	$-.0067$ (0/6)
Terminal selection	$+.0112$ (6/6)	$+.0128$ (6/6)	$+.0106$ (6/6)
Terminal $-$ short	$+.0190$ (6/6)	$+.0208$ (6/6)	$+.0173$ (6/6)

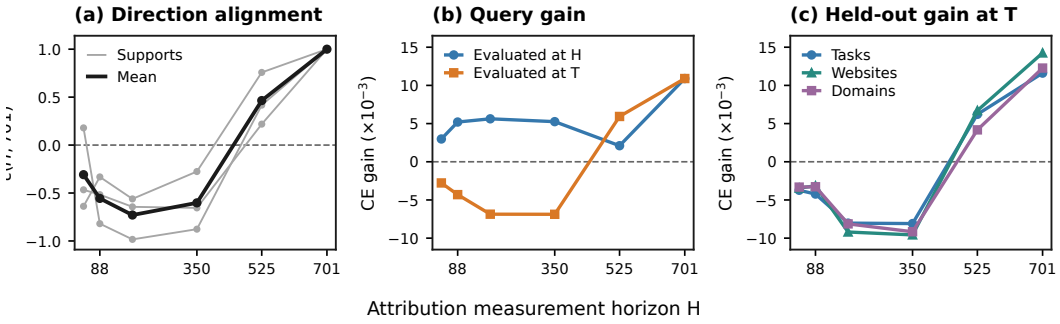

Figure 3: **Weighting transfer changes with the measurement horizon.** The first three Qwen supports are measured under the original selection seed at $H = 44, 88, 176, 350, 525, 701$, with all selected weights evaluated at $T = 701$. (a) Cosine similarity to the direction at T, estimated from four paired directions; gray lines are supports and black is their mean. (b) Mean query gain of the weighting chosen at H, evaluated at H or T. (c) Mean held-out gain of that weighting at T. Gains are uniform minus selected CE; positive is better. Lines connect measured points.

$+.0128$. It also improves 49/53 unseen-domain website means. Full website counts and descriptive support intervals appear in Appendix B.1. The terminal benefit therefore extends to data that did not select the weights.

This reversal accompanies a change in measured direction. All six Mind2Web supports have negative cosines between the short- and terminal-horizon directions (Figure 2a), with mean $-.600$ and range $[-.820, -.332]$. Short and terminal selection also choose different candidate weightings on every support. Appendix B.1 gives the six values.

Expanding the measurement from four to all six feature-space directions on the first three supports preserves negative short/terminal alignment. Both selection rules retain their original choices on every support (Appendix B.1.1).

5.2 Reversal persists across early measurement horizons

With candidates and full training paths fixed, failure extends beyond the original 88-update choice (Figure 3). At each of the four measured horizons from 44 to 350, selection lowers query loss at that horizon on 3/3 supports. Yet mean terminal gains are negative on the query and all three held-out panels. The mean cosine becomes more negative from 44 to 176, then recovers. Extending look-ahead within this early range does not recover positive mean terminal gains.

Later measurement recovers positive mean held-out gains. At 525 updates, all three cosines and mean held-out gains are positive, but the positive-support counts are still 2/3, 2/3, and 1/3 for tasks, websites, and domains. At 701 they reach 3/3 on every panel (Appendix B.1.3). Positive alignment and average recovery thus do not ensure beneficial finite choices on every support.

Table 2: Qwen3-VL-2B transfer of fixed weights to two new optimizer seeds on three Mind2Web supports. Values are uniform-minus-method action-token cross entropy at $T = 701$. The final column reports terminal minus short gain, computed before rounding. We average seeds within each support, then average the three support means. Parentheses count positive seed–support cells; these are nested within supports.

Panel	Short gain	Terminal gain	Terminal − short
Query	$-.0065$ (0/6)	$+.0104$ (6/6)	$+.0169$ (6/6)
Held-out task	$-.0051$ (0/6)	$+.0123$ (6/6)	$+.0174$ (6/6)
Unseen website	$-.0036$ (1/6)	$+.0147$ (6/6)	$+.0182$ (6/6)
Unseen domain	$-.0043$ (0/6)	$+.0136$ (6/6)	$+.0179$ (6/6)

Evaluation	Short selection	Terminal selection	Terminal − short	Short > 0	Terminal > 0
Query@K	$+.0046$	–	–	3/3	–
Query@T	$-.0027$	$+.0049$	$+.0076$	1/3	3/3
Held-out task	$-.0015$	$+.0045$	$+.0059$	1/3	3/3
Unseen website	$-.0026$	$+.0051$	$+.0077$	1/3	3/3
Unseen domain	$-.0030$	$+.0048$	$+.0077$	1/3	3/3

Table 3: InternVL3.5-2B replication on the first three Mind2Web supports. Entries are mean uniform-minus-rule action-token cross entropy, so positive is better. Short and terminal selection choose among the same eight candidate weightings at $K = 88$ and $T = 701$. The final columns count positive support effects. Terminal − short is the difference in gains, computed before rounding.

5.3 Reversal and terminal gains persist under new optimizer seeds

With the selected weights held fixed, both new training seeds reproduce the original ordering (Table 2; Figure 2d). In all six combinations of training subset and optimizer seed, short selection lowers query loss at K but raises it at T, while terminal selection lowers it at T. Terminal selection yields lower loss than short selection in every combination on the query and all three held-out panels. The descriptive interval for terminal-over-short query gain is wide and spans zero after averaging the two seeds within each support (Appendix B.1).

5.4 InternVL reproduces terminal gains with mixed direction alignment

InternVL favors terminal selection under the same Mind2Web data roles, first three supports, horizons, and candidate-set size (Table 3). Two of the three direction cosines are negative and one is positive. Short and terminal selection choose different candidates on all three supports. Short selection lowers the K query loss on 3/3 supports, but at T its mean gain is $-.0027$ and only one support remains positive. Terminal selection lowers the T query loss on 3/3 supports, with mean gain $+.0049$.

On held-out data, terminal selection yields lower loss than both uniform weighting and short selection on 3/3 supports in every panel (Table 3). Compared with short selection, it improves 44/53 unseen-domain website means and 86/100 trajectory means. Appendix B.2 gives every support and the descriptive intervals, which span zero on the held-out panels. The models agree on the benefit of terminal selection even though their cosine signs differ.

5.5 Android controls separate sign correction from direction combination

The Android controls examine when short-horizon weighting remains useful and whether correction requires changing the sign or combining directions.

In the first Android study, a useful weighting emerges after correcting the checkpoint direction's sign. The starting checkpoint-local direction requires no path measurement; executing it worsens both official panels (Table 4). One paired measurement corrects the sign, and combining four measured directions yields larger gains. Compared with short projection, terminal projection lowers IDD loss

Method	Meas. updates	IDD loss	Gain (%)	App-unseen loss	Gain (%)
Uniform	0	.715662	–	.687618	–
Raw local	0	.722370	−.937	.696390	−1.276
Sign correction	176	.715426	.033	.685676	.282
Short projection	704	.712673	.418	.683508	.598
Terminal projection	5608	.711977	.515	.683853	.548

Table 4: AndroidControl weighting comparisons, averaged over three fixed final-training seeds. IDD/app-unseen panels have 52/83 trajectories and 13/9 applications. Every final model trains for 701 updates. Measurement counts exclude common final training, feature construction, and evaluation. Raw local uses the checkpoint surrogate without a path measurement; sign correction uses the first paired direction at $K = 88$; projections use four paired directions at $K = 88$ or $T = 701$. Gain is relative loss reduction over uniform.

further but gives a slightly smaller gain on app-unseen data. Short-horizon projection therefore remains useful in this setting.

In the repeated-support Android follow-up, the candidate chosen by lowest query loss at T improves both fresh validation panels on all three supports (Appendix B.3). The query and validation banks also change, so this result does not isolate the effect of horizon.

Equal cumulative data weights can produce opposite loss changes under different schedules. The matched-budget executions and Adam counterexample in Appendix D demonstrate this effect when placement and batching change. This supports treating the schedule as part of the attribution setting and keeping it fixed in the horizon comparisons.

6 Discussion

Accurate evaluation at a short horizon need not identify weights that help after longer training. Our direct rule ranks candidates by measured query losses, so its reversal can occur without using a gradient estimate to choose the weights. This connects the dependence of influence on training time and optimizer state (Wang et al., 2025a; Lee et al., 2026; Deng et al., 2026; Liu & Li, 2026) to the quality of a weighting decision.

For data curation, a short-run selection should be judged by its effect at the intended training endpoint. This matters when short runs guide how a larger training budget is spent. Our Mind2Web comparisons show that the selection horizon affects generalization: choosing at T improves held-out expert-action loss under matched training settings.

Terminal measurements cost more training. The horizon sweep recovers positive mean gains before T, but benefits remain inconsistent across supports. A practical next step is to develop cheaper estimates of terminal utility, using final weighting effects to assess their quality. Appendix E reports the measurement and evaluation costs.

Limitations. The experiments cover designed weight families and rank-2 LoRA, using overlapping Mind2Web supports and Android subsets. Most measurements use four directions, with an extension to six on three Qwen supports; other direction seeds and radii remain untested. The horizon sweep reuses three supports and one seed. InternVL3.5-2B changes the multimodal stack while retaining a Qwen3-1.7B language backbone. The demonstrated benefit is lower offline expert-action cross entropy; autonomous evaluation is needed to establish gains in task completion.

7 Conclusion

Accurately measured short-horizon gains can lead to data weights that increase loss after continued training. Our paired experiments connect this failure to changes in attribution direction across training horizons. Selecting weights at the terminal horizon restores held-out expert-action loss gains in the studied GUI fine-tuning settings. These findings make the intended training endpoint an essential part of data-weight attribution.

AI USE STATEMENT

Generative AI tools provided feedback on experimental design and assisted with implementation and manuscript polishing.

REPRODUCIBILITY STATEMENT

Section 3 defines the weight family, horizon diagnostic, and selection rules. Appendix A specifies data roles, features, schedules, seeds, and training settings. The support-level appendix reports the Qwen3-VL, InternVL, and Android results; the final appendix details compute accounting. The anonymous repository is available at `https://anonymous.4open.science/r/horizon-matched-attribution-3C50/README.md`.

REFERENCES

Christopher Agia, Rohan Sinha, Jingyun Yang, Rika Antonova, Marco Pavone, Haruki Nishimura, Masha Itkina, and Jeannette Bohg. Cupid: Curating data your robot loves with influence functions, 2025.

Juhan Bae, Nathan Ng, Alston Lo, Marzyeh Ghassemi, and Roger B Grosse. If influence functions are the answer, then what is the question? volume 35, pp. 17953–17967, 2022.

Juhan Bae, Wu Lin, Jonathan Lorraine, and Roger Grosse. Training data attribution via approximate unrolling. volume 37, pp. 66647–66686, 2024.

Yuanyuan Chen, Boyang Li, Han Yu, Pengcheng Wu, and Chunyan Miao. Hydra: Hypergradient data relevance analysis for interpreting deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 7081–7089, 2021.

Krzysztof Choromanski, Mark Rowland, Vikas Sindhwani, Richard Turner, and Adrian Weller. Structured evolution with compact architectures for scalable policy optimization. In *International Conference on Machine Learning*, pp. 970–978. PMLR, 2018.

Rudrajit Das, Neel Patel, Meisam Razaviyayn, and Vahab Mirrokni. Less is more: Convergence benefits of fewer data weight updates over longer horizon. *arXiv preprint arXiv:2602.19510*, 2026.

Junwei Deng, Pingbang Hu, Suliang Jin, Hao Lu, Jiachen T Wang, Shichang Zhang, and Jiaqi W Ma. How faithful is trajectory-based data attribution? error sources, remedies, and practical guidelines. *arXiv preprint arXiv:2605.18814*, 2026.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. volume 36, pp. 28091–28114, 2023.

Logan Engstrom, Andrew Ilyas, Benjamin Chen, Axel Feldmann, William Moses, and Aleksander Madry. Optimizing ml training with metagradient descent. *arXiv preprint arXiv:2503.13751*, 2025.

Jinhui Guo. Delayed optimizer-state transport shapes short-horizon training decisions. *arXiv preprint arXiv:2608.24593*, 2026.

Yuzheng Hu, Pingbang Hu, Han Zhao, et al. Most influential subset selection: Challenges, promises, and beyond. volume 37, pp. 119778–119810, 2024.

Andrew Ilyas and Logan Engstrom. Magic: Near-optimal data attribution for deep learning, 2025.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pp. 1885–1894. Pmlr, 2017.

Jin Hwa Lee, Matthew Smith, Maxwell Adam, and Jesse Hoogland. Influence dynamics and stagewise data attribution. In *International Conference on Learning Representations*, volume 2026, pp. 35716–35747, 2026.

Wei Li, William Bishop, Alice Li, Chris Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. On the effects of data scale on ui control agents. volume 37, pp. 92130–92154, 2024.

Kang Liu and Suyan Li. Finite-horizon input-output dynamics of minibatch perturbations in adamw. *arXiv preprint arXiv:2608.19762*, 2026.

Niru Maheswaranathan, Luke Metz, George Tucker, Dami Choi, and Jascha Sohl-Dickstein. Guided evolutionary strategies: Augmenting random search with surrogate gradients. In *International Conference on Machine Learning*, pp. 4264–4273. PMLR, 2019.

Florian Meier, Asier Mujika, Marcelo Matheus Gauy, and Angelika Steger. Improving gradient estimation in evolutionary strategies with past descent directions. *arXiv preprint arXiv:1910.05268*, 2019.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. volume 33, pp. 19920–19930, 2020.

Amirreza Shaban, Ching-An Cheng, Nathan Hatch, and Byron Boots. Truncated back-propagation for bilevel optimization. In *The 22nd international conference on artificial intelligence and statistics*, pp. 1723–1732. PMLR, 2019.

Jiachen Tianhao Wang, Dawn Song, James Y Zou, Prateek Mittal, and Ruoxi Jia. Capturing the temporal dependence of training data influence. In *International conference on learning representations*, volume 2025, pp. 11236–11273, 2025a.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025b.

Yuhuai Wu, Mengye Ren, Renjie Liao, and Roger Grosse. Understanding short-horizon bias in stochastic meta-optimization. 2018.

Zheng Wu, Chengcheng Han, Zhengxi Lu, Tianjie Ju, Yanyu Chen, Qi Gu, Xunliang Cai, and Zhuosheng Zhang. Gui-cider: Mid-training gui agents via causal internalization and density-aware exemplar reselection. *arXiv preprint arXiv:2605.28534*, 2026.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. Less: Selecting influential data for targeted instruction tuning. 2024.

Yuxin Yang, Aoxiong Zeng, and Xiangquan Yang. The long-term effects of data selection in llm fine-tuning. *arXiv preprint arXiv:2605.30537*, 2026.

Yang Yu, Kai Han, Hang Zhou, Yehui Tang, Kaiqi Huang, Yunhe Wang, and Dacheng Tao. Llm data selection and utilization via dynamic bi-level optimization. 2025.

A Protocol and Reproducibility Details

A.1 Inputs, targets, and data roles

Each example contains a screenshot, task instruction, and at most four previous reference actions. The prompt asks for the next executable action. Android targets are compact JSON actions; Mind2Web targets retain the demonstrated human-readable Web action string. We apply the model chat template and compute cross entropy only on assistant action tokens; prompt and image tokens receive ignore labels. Feature construction uses training trajectories and selection-query gradients at the reference checkpoints specified below. Raw HTML, negative candidates, held-out data, and outcomes of the reweighted candidate paths are excluded.

Table 5 gives the trajectory counts for each data role. Mind2Web draws its training supports from a shared pool and uses the same query, warmup, and official evaluation panels across supports. Draws 0–5 identify the six supports; the last three were fixed as a consecutive extension before their model outcomes were observed. Data partitioning, support construction, direction construction, occurrence schedules, and training use fixed pseudorandom seeds.

Table 5: Trajectory counts by data role. The Android official intersections are subsets of the corresponding custom test panels, not additional disjoint sets. The three fresh Android query and validation sets are disjoint by trajectory.

Role	Trajectories	Detail
Mind2Web: Qwen3-VL-2B and InternVL3.5-2B		
Training pool	250	Shared source pool
Training support	25	Per draw from the pool
Selection query	60	Shared across supports
Warmup	1	Shared before the measured paths
Held-out task	100	`test_task`
Unseen website	100	`test_website`
Unseen domain	100	`test_domain`
AndroidControl: initial controlled intervention		
Training	500	Training pool
Selection query	100	Original query bank
Custom seen test	98	Seen applications
Custom unseen test	96	Unseen applications
Official in-distribution intersection	52	13 applications
Official app-unseen intersection	83	9 applications
AndroidControl: repeated-support follow-up		
Training support	25	Per draw from the 500-trajectory pool
Fresh selection query	100	New query bank
Fresh same-application validation	100	Seen applications
Fresh app-unseen validation	100	Unseen applications

On the first three supports, the selection seed supplies the short- and terminal-horizon choices. We keep those weights fixed and retrain the uniform and both selected-weight conditions under two additional seeds, labeled Seed A and Seed B. The selection seed is excluded from the transfer estimate.

The InternVL replication uses those same first three support identities and all original Mind2Web data roles. It reuses only the uniform occurrence streams and update boundaries. InternVL-specific checkpoint scores, feature whitening, directions, action-token counts, and query-based weighting choices are recomputed. Each support again uses nine paths with the same token budget and horizons as Qwen.

The controlled Android intervention evaluates the official intersections of its custom test panels. The repeated-support study uses three supports, labeled Draws 0–2, with fresh selection-query and validation banks (Table 5). Three final-training initializations are averaged within support.

A.2 Feature definitions and whitening

Table 6 gives every raw descriptor used to construct the intervention spaces. At checkpoint s, let $g_Q^{(s)}$ be the macro-averaged selection-query gradient over trainable parameters, and let $\Delta_i^{(s)}$ be the signed AdamW parameter update induced by trajectory i's token-mean loss gradient using the checkpoint's optimizer state. Each candidate update is simulated from that same state without advancing it. The checkpoint-direction utility is

$$v_i^{(s)} = -\frac{\langle g_Q^{(s)}, \Delta_i^{(s)} \rangle}{\|g_Q^{(s)}\| \, \|\Delta_i^{(s)}\|}.$$

Positive utility indicates a local reduction in query loss. The score compares the query gradient with an optimizer update, rather than two raw gradients. Mind2Web, including InternVL, uses the post-warmup utility. Android averages the utilities after warmup and after uniform training to T; its checkpoint-loss descriptor similarly averages their trajectory token-mean losses. Its structured-control fraction combines `select` and `set_range`; its scroll fraction combines all scroll action types. Mind2Web checkpoint loss is the trajectory token-mean loss and mean target tokens is averaged over its action steps.

Table 6: Trajectory descriptors used to construct each weight space.

Family	Raw trajectory descriptors
Mind2Web	checkpoint direction; log action-step count; checkpoint loss; mean action-target tokens; click, type, and select fractions
Android	checkpoint direction; log action-step count; checkpoint loss; click, type, scroll, long-press, and structured-control fractions

Let $X \in \mathbb{R}^{n \times p}$ be the raw feature matrix and $D_q = \mathrm{diag}(q)$ the base trajectory masses. We center each column within its weight-normalization group, set columns that are constant within all groups to zero, and divide active column k by $s_k = (\sum_i q_i \bar{X}_{ik}^2)^{1/2}$. Let Y denote the standardized matrix and

$$Y^\top D_q Y = V \Lambda V^\top. \tag{10}$$

We retain eigenvalues above $\epsilon_{64} \max(n, p) \lambda_{\max}$ and set

$$Z = Y V_r \Lambda_r^{-1/2}, \qquad Z^\top D_q Z = I. \tag{11}$$

The retained coordinates have identity covariance under the base weights. Mind2Web uses one global normalization group and yields rank six on each reported support. Android uses equal application mass and uniform mass within each application.

Let v_i be the checkpoint score defined above and set $a = -Z^\top D_q v$. The first direction is $a/\|a\|$; the remaining directions are fixed-seed Gaussian vectors orthogonalized by Gram–Schmidt until four orthonormal directions are obtained in the main experiments. The extension in Appendix B.1.1 continues the same construction to six. This construction, the support, and $h = .1$ are fixed before held-out evaluation.

A.3 Training configuration

The main Mind2Web and seed-transfer experiments use Qwen3-VL-2B. The replication uses `OpenGVLab/InternVL3_5-2B-HF`, with an InternViT vision stack and a Qwen3-1.7B language component. The Android study uses Qwen2.5-VL-3B. Qwen3 paths use SDPA, disable the model cache during training, and restrict image processing to 100,352–200,704 pixels. InternVL uses its native multimodal chat processor and language-side query/value LoRA. Within a comparison, the uniform and signed paths share their initialization, occurrence stream, batch boundaries, and learning rate schedule. Only the objective weights differ.

Each Mind2Web path (Qwen3 or InternVL) first replays the same warmup trajectory in action order, using one AdamW update per action's token-mean loss. The subsequent 701 updates retain the warmup parameters, moment estimates, and optimizer step counter. Warmup updates are excluded from the reported K/T horizons and token budget.

Table 7: Shared training configuration.

Component	Setting
Trainable parameters	LoRA on `q_proj`, `v_proj`; rank 2, $\alpha_{\mathrm{LoRA}} = 4$, dropout 0, no bias
Optimizer	AdamW; learning rate 2×10^{-5}, $\beta = (.9, .999)$, $\epsilon = 10^{-8}$, weight decay 0
Numerics	BF16 model/training dtype; one accelerator per path
Path length	701 measured updates after a common warmup; $K = 88$, $T = 701$
Budget	47,004 supervised target tokens per final path
Measurements	four orthonormal directions by default, paired radius $h = .1$
Projection	normalized secant direction with radius $\rho = .2$ when used
Randomness	fixed seeds; paired paths share all stochastic choices

A.4 Occurrence schedules and objective mass

For each Mind2Web support, the schedule reserves one occurrence of every selected action step. A systematic sampler then fills the remaining token budget from the application/trajectory/step macro distribution. The combined stream is shuffled once with the same seed and partitioned into 701 updates. If an action step appears r times, each occurrence receives one rth of that step's target objective mass. The normalized trajectory mass is therefore exact despite variable repetition. All paths for a support use the same occurrence sequence and batch boundaries. Their trajectory masses are changed only by applying the exponential tilt.

More precisely, let n_i be trajectory i's number of action steps, r_{ia} the total occurrences of its step a in the full schedule, and $\ell_{ia}(\theta)$ that step's mean supervised action-token loss. For the multiset of occurrences B_t at update t, backpropagation uses

$$J_t(\theta; \alpha) = T \sum_{(i,a) \in B_t} \frac{w_i(\alpha)}{n_i r_{ia}} \ell_{ia}(\theta).$$

Before this common scaling, occurrence masses sum to $w_i(\alpha)$ for each trajectory; its backpropagation coefficients sum to $T w_i(\alpha)$. The factor T makes the total gradient coefficient average one per update over the full schedule. We do not renormalize coefficients within an update, so between-update mass differences are retained. The same fixed terminal factor T applies before and after checkpoint K; a short-horizon measurement does not rescale the training prefix.

The six supports contain 175–211 distinct selected action steps. Their coverage reservation consumes 1,954–2,808 target tokens; systematic sampling fills the remainder to 47,004. Realized trajectory repetition ranges from 109 to 216 across supports. These are counts within each training stream.

A.5 Loss aggregation and selection order

Let ℓ_i be the mean cross entropy over the supervised action tokens of trajectory i, and let I_g be the evaluated trajectories in website or application g. The reported macro loss is

$$L_{\mathrm{macro}} = \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \frac{1}{|I_g|} \sum_{i \in I_g} \ell_i. \tag{12}$$

Mind2Web uses websites; Android uses applications. A positive gain is $L_{\mathrm{uniform}} - L_{\mathrm{method}}$. We score the K and T checkpoints of all signed paths on the same selection query: eight per support in the main experiment and twelve in the six-direction extension. Short and terminal selection then choose the candidate weighting with the lowest loss. Only after the choice is fixed do we score its T adapter and the matched uniform adapter on held-out panels.

Intervals are computed across support draws. For repeated Android supports, we first average the three training initializations within each support, giving three support means. The reported t-intervals describe these fixed small panels. Websites, applications, trajectories, and seed–support cells are nested within them and are not counted as independent replications.

Mind2Web seed transfer follows the same rule: average the two new seeds within each support and panel, then summarize the three support means. We also report the sign of each seed's effect. The seed used for selection is reported separately and excluded from these means and intervals.

B Full Support-Level Results

Table 8 collects the mean effects and descriptive intervals for Qwen, optimizer-seed transfer, and InternVL. The following subsections give the individual support results.

Table 8: Mean gains and descriptive intervals across Mind2Web support draws. Short and terminal gains are uniform minus selected loss; terminal − short compares the two selections. Evaluation is at T unless marked otherwise. The seed-transfer summary first averages the two new seeds within each support. Intervals describe the fixed panels; supports can overlap and share evaluation data.

Panel	Comparison	Mean gain	Interval	Positive supports
Qwen3-VL-2B: six supports, selection seed				
Query@K	Short	$+.004495$	$[.003181, .005808]$	6/6
Query@T	Short	$-.007306$	$[-.014031, -.000581]$	0/6
Query@T	Terminal	$+.009464$	$[.003077, .015850]$	6/6
Query@T	Terminal − short	$+.016770$	$[.006854, .026686]$	6/6
Held-out task	Terminal	$+.011177$	$[.005325, .017029]$	6/6
Unseen website	Terminal	$+.012839$	$[.007061, .018617]$	6/6
Unseen domain	Terminal	$+.010643$	$[.006085, .015201]$	6/6
Qwen3-VL-2B: three supports, two new optimizer seeds				
Query@T	Terminal − short	$+.016901$	$[-.003680, .037483]$	3/3
InternVL3.5-2B: three supports				
Query@T	Terminal	$+.004908$	$[.001241, .008576]$	3/3
Held-out task	Terminal	$+.004458$	$[-.001013, .009929]$	3/3
Unseen website	Terminal	$+.005120$	$[-.000544, .010784]$	3/3
Unseen domain	Terminal	$+.004760$	$[-.000206, .009725]$	3/3
Held-out task	Terminal − short	$+.005944$	$[-.003635, .015523]$	3/3
Unseen website	Terminal − short	$+.007720$	$[-.006433, .021873]$	3/3
Unseen domain	Terminal − short	$+.007747$	$[-.006741, .022235]$	3/3

B.1 Mind2Web

Table 10 reports the direction cosines and query effects; Table 11 reports held-out gains. Table 9 summarizes how many websites and trajectories benefit under each comparison.

Table 9: Mind2Web websites and trajectories with positive mean effects. Terminal compares terminal selection with uniform; terminal − short compares the two selections. Counts at the two levels summarize the same evaluation panels and are not independent replications.

Panel	Comparison	Websites	Trajectories
Qwen3-VL-2B: six supports			
Held-out task	Terminal	23/23	88/100
Unseen website	Terminal	10/10	89/100
Unseen domain	Terminal	49/53	87/100
Held-out task	Terminal − short	22/23	93/100
Unseen website	Terminal − short	10/10	91/100
Unseen domain	Terminal − short	48/53	92/100
InternVL3.5-2B: three supports			
Unseen domain	Terminal − short	44/53	86/100

Table 10: Qwen3-VL-2B Mind2Web selection-query results by support. Gains are uniform minus selected-model action-token cross entropy. Short@K selects and is evaluated at $K = 88$; the last two columns evaluate the short- and terminal-selected weightings at $T = 701$.

Support	$c(K, T)$	Short@K	Short@T	Terminal@T
Draw 0	$-.516904$	$+.004761$	$-.002566$	$+.003604$
Draw 1	$-.331987$	$+.005516$	$-.001930$	$+.019248$
Draw 2	$-.819537$	$+.005331$	$-.008413$	$+.009846$
Draw 3	$-.813406$	$+.003919$	$-.010669$	$+.005069$
Draw 4	$-.537210$	$+.005219$	$-.002209$	$+.005287$
Draw 5	$-.580002$	$+.002223$	$-.018051$	$+.013730$

Table 11: Qwen3-VL-2B Mind2Web held-out gains by support. Each split shows short selection (S) and terminal selection (T). Columns are website-macro action-token cross entropy gains over the support-matched uniform path.

	Held-out task		Unseen website		Unseen domain	
Support	S	T	S	T	S	T
Draw 0	$-.002216$	$+.005100$	$+.001715$	$+.006488$	$-.001167$	$+.006724$
Draw 1	$-.005440$	$+.018934$	$-.005802$	$+.020234$	$-.003530$	$+.015657$
Draw 2	$-.005027$	$+.010789$	$-.005367$	$+.016020$	$-.005119$	$+.014427$
Draw 3	$-.013731$	$+.011296$	$-.012584$	$+.009850$	$-.007706$	$+.006896$
Draw 4	$-.002526$	$+.005123$	$-.004311$	$+.007934$	$-.003266$	$+.006555$
Draw 5	$-.018029$	$+.015821$	$-.021654$	$+.016506$	$-.019109$	$+.013598$

B.1.1 Number of measurement directions

To test whether the negative alignment depends on measuring only part of the feature space, we extend the first three Qwen supports from four to six orthonormal directions spanning the same six-dimensional feature space. We retain the original four directions and 27 training paths, and add two paired directions per support, requiring 12 new paths. The supports, feature coordinates, radius $h = .1$, random settings, schedules, token budget, and horizons $K = 88$ and $T = 701$ are unchanged.

All three cosines remain negative (Table 12). Selection over the expanded set of 12 signed candidates retains the original short and terminal choices on every support. Their query reversal and held-out gains therefore remain unchanged. We reuse the original held-out evaluations of these same adapters. Finite-radius measurements and discrete choices can still depend on the basis.

Table 12: Extending the first three Qwen supports from four to six measurement directions. Short/terminal direction cosines remain negative, and neither selection rule changes its chosen weighting.

Support	Cosine, $m = 4$	Cosine, $m = 6$	Short choice	Terminal choice
Draw 0	$-.517$	$-.578$	Unchanged	Unchanged
Draw 1	$-.332$	$-.270$	Unchanged	Unchanged
Draw 2	$-.820$	$-.775$	Unchanged	Unchanged

B.1.2 Transfer of fixed weights to new optimizer seeds

Table 13 gives every combination of support and new optimizer seed. The weights are not reselected. Short selection lowers query loss at K but raises it at T in all six combinations. At T, terminal selection yields lower query loss than both uniform weighting and short selection.

The summary row in Table 13 averages the two seeds within each support. The interval across those support means appears in Table 8.

Table 13: Mind2Web transfer of fixed weights, by seed and support. All values are loss reductions; positive values favor the named method. Short@K is uniform minus short loss at update 88. Short@T and terminal@T are uniform minus method loss at update 701. The remaining columns are terminal minus short gain at update 701. Seed A and Seed B denote the two additional training seeds; the selection seed is excluded.

Support	Seed	Short@K	Short@T	Terminal@T	Query $T-S$	Task $T-S$	Website $T-S$	Domain $T-S$
Draw 0	A	+.006659	−.002654	+.003865	+.006519	+.007811	+.004838	+.010723
Draw 0	B	+.004959	−.007023	+.001238	+.008261	+.009607	+.005585	+.010509
Draw 1	A	+.003935	−.005758	+.016156	+.021913	+.022352	+.025315	+.020904
Draw 1	B	+.002029	−.005224	+.017960	+.023184	+.026108	+.029118	+.022256
Draw 2	A	+.005590	−.007840	+.012405	+.020245	+.018545	+.022113	+.022498
Draw 2	B	+.005529	−.010331	+.010955	+.021285	+.019863	+.022448	+.020699

Mean over Seeds A/B	Draw 0	Draw 1	Draw 2
Query $T-S$	+.007390	+.022549	+.020765

B.1.3 SIX-HORIZON MEASUREMENTS

The sweep reuses the first three supports, the selection seed, the original four directions, $h = .1$, and all 27 fixed training paths. Each path is replayed once to 701, with query checkpoints at 44, 88, 176, 350, 525 and 701. Extra query measurements preserve Python, NumPy and Torch CPU/CUDA random states. The full-T objective normalization is unchanged at every checkpoint. All 54 replayed query losses at the original 88 and 701 checkpoints match their historical values exactly.

At each H, Eq. (8) selects among the same eight candidates. Choices are made from the selection query before evaluating held-out panels; no preferred H is chosen from held-out outcomes. Repeated selections share the same terminal adapter evaluation. Table 14 reports all support-level values, and Table 15 reports the selected coordinates and positive-support counts. The six measurement times are correlated observations on three supports.

Table 14: Six-horizon sweep on the first three Qwen supports. Gains are uniform minus selected CE ($\times 10^{-3}$). Query@H evaluates the weighting selected at H at that same time; all other gains evaluate the same weights at $T = 701$.

Support	H	$c(H,T)$	Query@H	Query@T	Task@T	Website@T	Domain@T
Draw 0	44	−0.466	+4.249	−7.326	−7.734	−7.245	−7.587
Draw 0	88	−0.517	+4.761	−2.566	−2.216	+1.715	−1.167
Draw 0	176	−0.644	+5.964	−2.566	−2.216	+1.715	−1.167
Draw 0	350	−0.655	+4.090	−2.604	−2.387	+0.577	−4.251
Draw 0	525	+0.219	+3.118	−2.566	−2.216	+1.715	−1.167
Draw 0	701	+1.000	+3.604	+3.604	+5.100	+6.488	+6.724
Draw 1	44	−0.638	+1.636	−2.308	−4.710	−1.338	−0.863
Draw 1	88	−0.332	+5.516	−1.930	−5.440	−5.802	−3.530
Draw 1	176	−0.561	+6.205	−1.930	−5.440	−5.802	−3.530
Draw 1	350	−0.276	+6.888	−1.930	−5.440	−5.802	−3.530
Draw 1	525	+0.758	+2.076	+19.248	+18.934	+20.234	+15.657
Draw 1	701	+1.000	+19.248	+19.248	+18.934	+20.234	+15.657
Draw 2	44	+0.179	+3.066	+1.319	+1.257	−1.742	−1.502
Draw 2	88	−0.820	+5.331	−8.413	−5.027	−5.367	−5.119
Draw 2	176	−0.983	+4.703	−16.097	−16.448	−23.504	−19.702
Draw 2	350	−0.877	+4.781	−16.097	−16.448	−23.504	−19.702
Draw 2	525	+0.417	+1.114	+1.115	+1.918	−1.835	−1.957
Draw 2	701	+1.000	+9.846	+9.846	+10.789	+16.020	+14.427

For Draw 0, the 88-, 176- and 525-update rules select the same candidate, so their terminal gains coincide despite different query losses at selection. At 525, this support has positive cosine but negative terminal query gain. Draw 1 already selects its terminal candidate at 525; Draw 2 does not. Mean gains at an intermediate horizon can therefore hide losses on individual supports.

Table 15: Selected candidate weightings and counts of supports with positive terminal held-out gains. Candidates are $\pm hu_j$, with $h = .1$ and $j = 1, \ldots, 4$. Counts are reported separately for tasks, websites and domains.

H	Draw 0	Draw 1	Draw 2	Task positive	Website positive	Domain positive
44	$+hu_2$	$+hu_2$	$-hu_4$	1/3	0/3	0/3
88	$+hu_1$	$+hu_1$	$+hu_1$	0/3	1/3	0/3
176	$+hu_1$	$+hu_1$	$-hu_3$	0/3	1/3	0/3
350	$+hu_4$	$+hu_1$	$-hu_3$	0/3	1/3	0/3
525	$+hu_1$	$+hu_3$	$-hu_2$	2/3	2/3	1/3
701	$-hu_2$	$+hu_3$	$+hu_3$	3/3	3/3	3/3

B.2 INTERNVL3.5-2B

Tables 16 and 17 expand Table 3. No new optimizer seeds are included in this replication.

Table 16: InternVL3.5-2B selection-query results by support. Gains are uniform minus selected-model action-token cross entropy. Short@K is evaluated at $K = 88$; both final columns are evaluated at $T = 701$.

Support	$c(K, T)$	Short@K	Short@T	Terminal@T
Draw 0	$+.189392$	$+.001803$	$+.003201$	$+.004203$
Draw 1	$-.173678$	$+.003343$	$-.009401$	$+.006605$
Draw 2	$-.282366$	$+.008790$	$-.001815$	$+.003917$

Table 17: InternVL3.5-2B held-out gains by support. S and T denote short and terminal selection, evaluated at $T = 701$. Entries are website-macro action-token cross entropy gains over the support-matched uniform path.

	Held-out task		Unseen website		Unseen domain	
Support	S	T	S	T	S	T
Draw 0	$+.001067$	$+.005057$	$+.004112$	$+.007395$	$+.000976$	$+.005714$
Draw 1	$-.004087$	$+.006298$	$-.009016$	$+.005129$	$-.008367$	$+.006103$
Draw 2	$-.001438$	$+.002018$	$-.002897$	$+.002835$	$-.001572$	$+.002463$

InternVL's held-out 95% t-intervals in Table 8 span zero despite positive effects on each support.

B.3 ANDROIDCONTROL

The terminal single-axis rule uses $-\rho \operatorname{sign}(\widehat{g}_{T,1})u_1$ with $\rho = .2$. We also multiply the full-query terminal projection by $\sqrt{(1 + \cos(\widehat{d}_T^{(1)}, \widehat{d}_T^{(2)}))/2}$, where the two directions are measured on the selection query's two folds. The factor uses only query measurements and shrinks the projection when the folds disagree.

Table 18: Terminal-horizon calibration on fresh AndroidControl roles. Entries are application-macro loss gains over uniform (uniform minus candidate); positive is better. The last two columns count positive support means out of three after averaging three training seeds within support.

Method	Fresh query	Seen val.	App-unseen val.	Seen	App-unseen
Terminal single axis	$-.000061$	$-.000406$	$-.000167$	0/3	1/3
Terminal projection	$+.001086$	$+.000679$	$+.001140$	2/3	2/3
Fold-scaled projection	$+.000992$	$+.000576$	$+.001043$	3/3	2/3
Terminal direct	$+.000413$	$+.000296$	$+.000555$	3/3	3/3

Earlier short-calibrated candidates increased the original T selection-query loss on every support. The follow-up changes the query and validation banks as well as the measurement horizon. Direct

selection lowers fresh-query loss and both validation losses on all three supports. Table 19 expands the repeated-support result reported in Table 18. The direct rule is the only one positive on both validation roles for every support. Projection has the largest mean held-out gains but mixed support signs; the single-axis rule is negative on every seen-validation support.

Table 19: Android terminal-calibration gains by support after averaging three final-training initializations within each support. Values are application-macro action-token cross entropy gains over matched uniform.

	Same-application validation			App-unseen validation		
Method	Draw 0	Draw 1	Draw 2	Draw 0	Draw 1	Draw 2
Single axis	$-.000451$	$-.000350$	$-.000416$	$-.000552$	$+.000741$	$-.000690$
Projection	$-.000354$	$+.001885$	$+.000508$	$-.000398$	$+.003278$	$+.000541$
Fold-scaled projection	$+.000034$	$+.001495$	$+.000200$	$-.000560$	$+.002964$	$+.000724$
Direct	$+.000199$	$+.000633$	$+.000056$	$+.000414$	$+.000754$	$+.000497$

C Finite-Radius Interpretation

Derivative limits refer to an idealized differentiable training map; the reported experiments use BF16 training at a fixed measurement radius.

Equation (5) describes a local projected direction, not the finite candidates used by Eq. (8). At $h = .1$, positive cosine can coexist with a loss increase, and negative cosine with a beneficial coordinate choice. Direct selection therefore compares the trained candidate losses themselves. Projection creates a new radius-ρ point beyond those measured candidates and requires a separate training run to evaluate its loss.

D Schedule Counterexample

Cumulative weights do not specify when samples contribute to training. Under adaptive optimization, gradients and moment states depend on earlier updates. Two schedules with the same total trajectory weights can therefore reach different final parameters. We compare uniform mass on the original schedule (U), random support mass on that schedule (M), and the same random mass on its repeated support schedule (R). M and R have identical cumulative target mass, 701 updates, and 47,004 processed target tokens. They differ in the placement and batching of positive-mass occurrences.

Table 20: Schedule counterexample. Lower application-macro action-token cross entropy is better. M and R have the same cumulative random-support objective mass, 701 updates, and 47,004 processed target tokens, but realize that mass on different occurrence schedules.

Recipe	Official IDD	Official unseen	Custom seen	Custom unseen
U: uniform, original schedule	.715662	.687618	.700722	.688821
M: random mass, original schedule	1.129974	1.057599	1.121251	1.065800
R: random mass, repeated schedule	.709048	.672496	.693004	.673368

In M, only 2,470 processed target tokens carry positive objective weight. Its 133 positive-mass updates are interleaved with 568 zero-mass updates. R supplies positive supervision throughout its repeated-support stream. Equal processed token counts therefore do not imply equal effective supervision. The M–R contrast changes repetition, grouping, timing, and Adam state together; it establishes schedule sensitivity, not a causal claim about repetition alone.

A scalar Adam counterexample shows why cumulative mass is insufficient even without model nonlinearity. With zero initial moments, constant learning rate, zero weight decay, $\epsilon = 0$, and scalar $g \neq 0$, the sequences (g, g) and $(2g, 0)$ have the same summed gradient. Under bias-corrected Adam, the parameter displacement is $-2\eta\,\mathrm{sign}(g)$ for the first sequence and

$$-\eta \left[1 + \frac{\beta_1}{1+\beta_1}\sqrt{\frac{1+\beta_2}{\beta_2}}\right] \mathrm{sign}(g) \tag{13}$$

for the second. At $(\beta_1, \beta_2) = (.9, .999)$, the displacement magnitudes are 2η and 1.6701η, respectively. This schedule dependence motivates including σ in the training recipe τ in Eq. (3).

E Compute Accounting

For direct selection over m paired directions, a run that selects at K can stop $2m - 1$ paths at K and continue the selected path. Its training cost, excluding warmup, feature construction and evaluation, is

$$C_{\text{short}} = 2mK + (T - K) = T + (2m - 1)K. \tag{14}$$

Terminal selection evaluates all candidate weightings after the full run, so $C_{\text{terminal}} = 2mT$. At $m = 4$, $K = 88$, and $T = 701$, the two costs are 1,317 and 5,608 optimizer updates, respectively. Relative to one T-update path, they are $1.88\times$ and $8.00\times$. A uniform comparator adds one path to an experiment but is not required to return the selected weighting. Under the training seed used for selection, direct selection also requires no separate final-model run: the chosen adapter is one of the measured paths. Projection adds a new T-update path because its combined coefficient vector was not one of the measured candidate weightings.

At $H = 525$, the same accounting gives 4,376 updates, 22% fewer than the 5,608 updates for terminal selection. This is a hypothetical cost if unselected paths stop at H: the reported sweep trained every candidate to T, and held-out gains at 525 still had mixed signs across supports.

Table 21 reports recorded compute by experiment and timing basis. Training was organized into parallel batches using up to eight GPUs, with one GPU per path. Mind2Web runs used GH200 GPUs. Application-recorded runtimes exclude preparation and scheduler overhead. The horizon sweep's training runtime includes extra query measurements and all 27 full paths. A separate implementation check added less than 0.01 GPU-hours and contributes no results.

Table 21: Recorded compute in GPU-hours. Allocated time and application-recorded runtime are grouped separately. Dashes indicate that no separate breakdown is reported, not zero cost. Android's combined value covers measurement, construction diagnostics, final training, and evaluation. Failed and cancelled runs contribute no results.

Experiment	Training	Evaluation	Other / combined
Allocated GPU-hours			
Qwen Mind2Web, six supports	14.290	0.860	–
Fixed-weight seed transfer	4.733	0.845	–
Six-direction evaluation retrieval	–	0.2883	–
Android terminal calibration	–	–	21.6325
Failed evaluator	–	0.023	–
Cancelled launch	–	–	0.162
Application-recorded GPU-hours			
InternVL, three supports	10.008	0.574	–
Qwen six-horizon sweep	8.524	0.629	–
Qwen six-direction extension	3.0784	0	–

The six-direction extension adds 12 full training paths and reuses the original 27. Its selected adapters are unchanged, so the evaluation entries were retrieved from the original measurements without new forward passes. The retrieval jobs still account for 0.2883 allocated GPU-hours; this is separate from the 3.0784 application-recorded training GPU-hours.