LEARNING TO SEE FROM READING: CROSS-MODAL SELF-DISTILLATION FOR CODING

Anonymous authors
Paper under double-blind review

ABSTRACT

A multimodal model can solve the same problem differently depending on whether the input is presented as text or as an image, even when both views preserve the same content. This asymmetry suggests a natural question: *can the model use its stronger view to improve its weaker one?* We study this question through same-model cross-modal on-policy distillation (OPD) for coding from rendered images. Specifically, an image-conditioned student generates its own responses, while a frozen copy of the same pretrained model reads the paired source text and provides next-token distributions along the student's prefixes. Matching these distributions transfers supervision from the stronger text view without ground-truth solutions, task rewards, or a stronger external teacher. We further characterize this mechanism through *projected view repair*, which separates raw cross-modal disagreement from the component expressible by the student's adaptation tangent and its alignment with task improvement. Empirically, mean image-input accuracy on MBPP+ rises from 46.6% to 52.6%, outperforming ground-truth supervised fine-tuning. Matched controls show that text-conditioned feedback yields a stronger image-student endpoint than image-conditioned feedback, while exact transcription also improves across rendering densities. The gains extend across views: image-trained checkpoints improve text-input coding beyond the frozen pretrained teacher. Together, these results show that a model can use a stronger representation of the same task as a source of self-supervision for a weaker one.

1 INTRODUCTION

A multimodal model can know how to solve a problem without accessing that knowledge equally well from every input representation. This distinction between *knowledge* and *access* is especially clear in coding: the same function specification may be provided as source text, a screenshot, or a terminal capture, while a single visual misread can change whether the generated program executes correctly. In our evaluation, the same pretrained Qwen3.5-4B model solves 444 of 591 tasks from text but only 326 from rendered images, a 19.9-point gap with the underlying task content held fixed (Fig. 1(a)). Importantly, this gap need not mean that the image-conditioned model lacks the relevant programming knowledge; the stronger text view shows that the knowledge is already present in the shared model. This creates a supervision opportunity: rather than importing a stronger external teacher, can the model use its own stronger representation of the same task to improve how it accesses that knowledge from a weaker view?

This opportunity matters as coding agents increasingly operate through screenshots and graphical interfaces (Yang et al., 2025b; Xie et al., 2024), while document models and optical context compression make rendered text a practical model input (Lee et al., 2023; Wei et al., 2025; Cheng et al., 2026). Prior work shows that equivalent content can produce substantially different model behavior across representations (Fu et al., 2024), and recent methods use stronger supervision to bridge such gaps. Reading, Not Thinking transfers text-mode reasoning traces (Sun et al., 2026), CAPS combines text-history feedback with reinforcement learning (Fan et al., 2026), and Self-Distilled Reasoner privileges a same-model teacher with solutions (Zhao et al., 2026). These methods strengthen the information available to the teacher. Our setting asks a different question: *is the stronger representation itself enough to provide useful supervision, even when the teacher receives no additional solution, reward, or external capability?*

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.

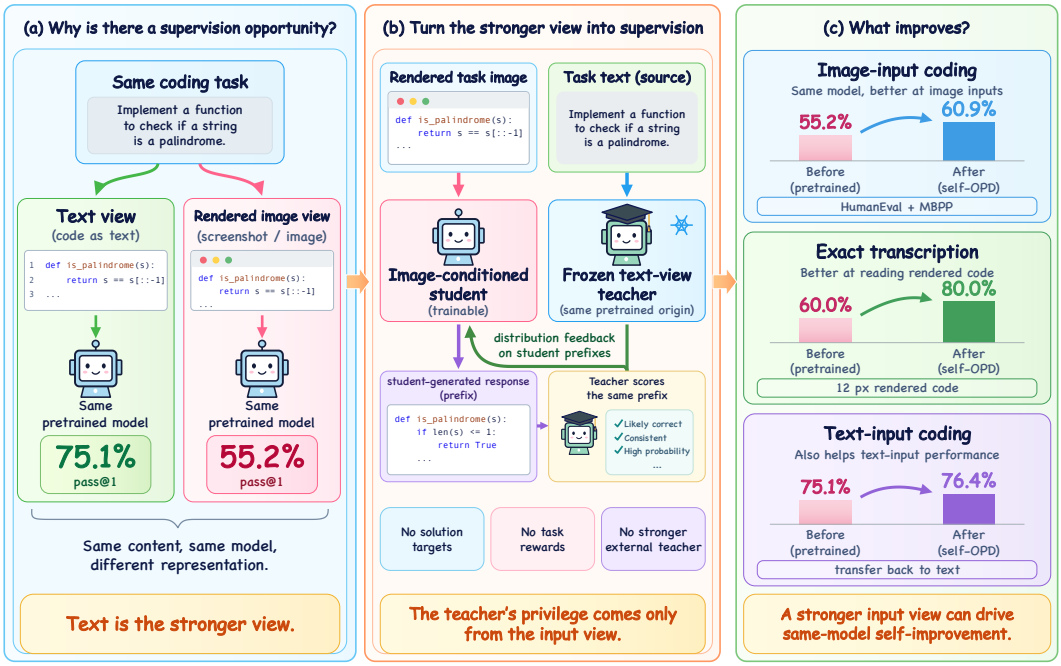

Figure 1. **Learning from a stronger view.** (a) The pretrained model achieves 75.1% text-input and 55.2% image-input coding accuracy. (b) A frozen text-view copy provides distribution feedback on the image student's own response prefixes. (c) The resulting checkpoints improve image coding, exact transcription, and text coding. Paired training inputs preserve the underlying source content.

We introduce *cross-modal self-OPD*, which turns this representation asymmetry into same-model supervision. Specifically, an image-conditioned student generates its own response, while a frozen copy of the same pretrained model reads the paired source text and scores the same generated prefixes (Fig. 1(b)). The student then matches the teacher's next-token distributions. Crucially, the teacher receives no solution target, task reward, or stronger external capability; its only privilege is access to the stronger text view of the same task. We use a two-stage training schedule. The first stage refreshes on-policy responses as the student changes (Agarwal et al., 2024), while the second replays a fixed response bank collected from the first-stage endpoint. The two stages combine adaptive coverage of the student's current prefixes with stable, task-focused coding adaptation.

The central insight is that *not every cross-modal disagreement is useful*. A text–image prediction difference can improve the image student only if the corresponding correction is expressible by the trainable adapter and the resulting update is aligned with task performance. We formalize this with a theory of *projected view repair*. Locally, the JSD objective becomes a Fisher-weighted projection of the text–image residual onto the image-conditioned LoRA tangent, separating repairable disagreement from components the adapter cannot express. This view also explains why additional teacher context need not help: it can enlarge disagreement while rotating the useful component away from the student's trainable or task-aligned directions. Because the adapter modifies shared language layers, the same update can also change text-conditioned behavior. Finally, token-margin conditions connect these continuous corrections to discrete outcomes such as exact transcription and executable programs.

The empirical results support this account across both input views (Fig. 1(c)). Across three complete runs, mean image-input pass@1 increases from 55.2% to 60.9%, while the stricter MBPP+ evaluation retains a 6.0-point gain. Matched image-student controls further isolate the source of supervision: a text-conditioned teacher produces a stronger endpoint than an image-conditioned teacher, and providing task text alone outperforms augmenting the teacher with a ground-truth solution. The improvement is not limited to coding accuracy. Exact transcription increases at both rendering densities, indicating better access to visually presented code, while all three image-trained checkpoints also surpass the frozen teacher on text-input coding. Thus, cross-modal self-distillation does more than repair the weaker image pathway: updates learned from the text–image asymmetry can improve behavior through the model's shared representation across views.

We make three contributions:

- We introduce cross-modal self-OPD, a same-model distillation framework that turns a model's stronger input view into supervision for its weaker one, without ground-truth solutions, task rewards, or a stronger external teacher.
- We develop a theory of projected view repair that separates cross-modal disagreement from the component expressible by the student's adaptation tangent, characterizes when this correction aligns with task improvement, and explains how the update can transfer across input views.
- We show empirically that cross-modal self-OPD improves image-input coding and exact transcription, that text-conditioned feedback is more effective than matched image-conditioned feedback, and that image-trained checkpoints can also improve text-input coding beyond the frozen teacher.

2 Related Work

Distillation on student-generated responses. Knowledge distillation transfers output distributions (Hinton et al., 2015), while sequence-level distillation trains on teacher samples (Kim & Rush, 2016). A central challenge is the mismatch between supplied training prefixes and the learner's own inference-time tokens (Bengio et al., 2015; Ranzato et al., 2016). Querying an expert on learner trajectories addresses this mismatch in imitation learning (Ross et al., 2011); ImitKD extends the idea to sequence distillation (Lin et al., 2020), and generalized knowledge distillation supports token-level divergences such as the Jensen–Shannon objective used here (Agarwal et al., 2024). Recent on-policy distillation also transfers reasoning from larger models (Yang et al., 2025a; Lu & Thinking Machines Lab, 2025). We instead study on-policy feedback between copies of the same pretrained model conditioned on different input views.

Privileged inputs and same-model teachers. Learning using privileged information gives a teacher a training-time information advantage (Vapnik & Vashist, 2009; Lopez-Paz et al., 2016), while context and self-distillation methods privilege the teacher with prompts, solutions, feedback, demonstrations, documents, or consensus (Askell et al., 2021; Snell et al., 2022; Zhao et al., 2026; Hubotter et al., 2026; Shenfeld et al., 2026; Stein et al., 2026; Gkountouras et al., 2026). Our teacher instead receives only the source-text representation of the student's rendered task, preserving problem content while changing its view. Unlike earlier cross-modal supervision between separate networks (Gupta et al., 2016), our teacher and student share the same pretrained origin, and supervision is applied on the student's own prefixes. Our theory further separates privileged-view disagreement from the portion expressible by the student's adaptation tangent.

Reading and coding from pixels. Prior work documents representation-dependent performance (Fu et al., 2024; Zhang et al., 2024; Rahmanzadehgervi et al., 2024), while pixel-based language models, OCR-free document systems, and optical context compression make rendered text a practical model input (Rust et al., 2023; Kim et al., 2022; Lee et al., 2023; Wei et al., 2025; Cheng et al., 2026). Visual coding benchmarks span plots, web pages, programming problems, and software interfaces (Wu et al., 2025; Yun et al., 2024; Li et al., 2024; Yang et al., 2025b; Xie et al., 2024). We instead render standard coding problems to hold task content fixed across views and evaluate generated programs by execution (Chen et al., 2021; Austin et al., 2021; Liu et al., 2023).

Different-view self-distillation. Recent methods create teacher advantages through informative crops, clean views, or generated visual thoughts (Yuan et al., 2026; Li et al., 2026b;a). Across the text–image boundary, Reading, Not Thinking uses text-mode reasoning traces (Sun et al., 2026), while CAPS combines a frozen text-history teacher with reinforcement learning (Fan et al., 2026). Our method instead uses the pretrained model's text-view next-token distributions directly, without task rewards or solution targets, and evaluates image improvement, transfer back to text, and changes in teacher context separately.

3 Cross-Modal Self-Distillation

3.1 One pretrained model, two task views

A task consists of source content x and an instruction u. A deterministic renderer R converts the same content into one or more image pages without changing the underlying task. As shown in Fig. 2(a), the student receives $(R(x), u)$, while the frozen teacher receives (x, u). Both start from the

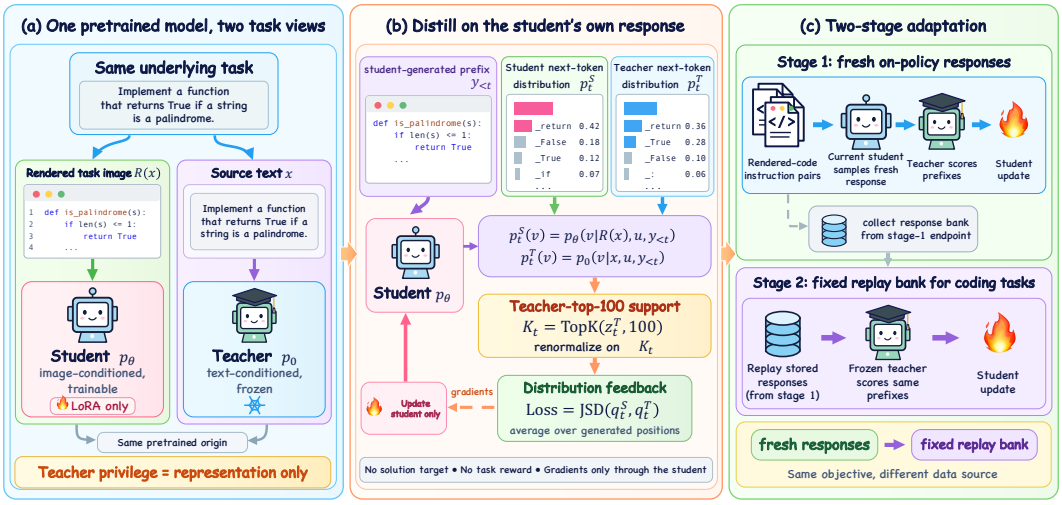

Figure 2. Cross-modal self-OPD. (a) Paired image and text views condition a LoRA student and a frozen teacher with the same pretrained weights. (b) Both score the student's generated prefixes, and teacher-top-100 renormalized JSD updates only the student. (c) Fresh on-policy sampling is followed by fixed replay of responses collected from the first-stage endpoint.

same pretrained model p_0; their supervision asymmetry comes from the input view rather than from a stronger model.

The student adapts the language-model attention and feed-forward projections with LoRA (Hu et al., 2022), while the pretrained weights, vision encoder, and projector remain fixed. Each adapted matrix is

$$W = W_0 + (\alpha/r)BA,$$

where A and B are trainable factors, r is the rank, and α is the scaling factor. With the adapters disabled, student and teacher share the same pretrained weights and differ only in their input representation.

At a student-generated prefix $y_{<t}$, the two views induce next-token distributions

$$p_t^S(v) = p_\theta(v \mid R(x), u, y_{<t}), \qquad p_t^T(v) = p_0(v \mid x, u, y_{<t}). \tag{1}$$

The same generated response IDs are appended after each model's own prompt, so the comparison is aligned by response-token position rather than by absolute sequence position. The final prompt logit predicts the first response token, and each subsequent logit predicts the next generated token. Appendix A.1 gives the prompt templates and rendering geometry.

3.2 Distribution feedback on student-generated prefixes

The stronger text view becomes supervision by evaluating the student's own trajectory rather than supplying a separate target response. Fig. 2(b) shows this update. Let $T(y)$ denote the positions corresponding to generated response tokens. At each position, the teacher selects its top-100 vocabulary items,

$$K_t = \text{TopK}(z_t^T, \min(100, |V|)), \tag{2}$$

$$q_t^S(v) = \frac{\exp z_t^S(v)}{\sum_{w \in K_t} \exp z_t^S(w)}, \qquad q_t^T(v) = \frac{\exp z_t^T(v)}{\sum_{w \in K_t} \exp z_t^T(w)}, \quad v \in K_t. \tag{3}$$

Thus, both distributions are compared on a teacher-anchored support while preserving their relative probabilities over the same alternatives.

With $m_t = (q_t^S + q_t^T)/2$, the response-level objective is

$$\mathcal{L}(\theta; x, y) = \frac{1}{|T(y)|} \sum_{t \in T(y)} \left[\tfrac{1}{2}\text{KL}(q_t^S \| m_t) + \tfrac{1}{2}\text{KL}(q_t^T \| m_t)\right]. \tag{4}$$

This is equal-weight Jensen–Shannon divergence (JSD) (Lin, 1991), following the token-level distribution-matching formulation of generalized knowledge distillation (Agarwal et al., 2024). Selected logits are computed in Float32 with natural logarithms. Generated token IDs and teacher predictions are treated as fixed during each update, so gradients flow only through the student. Positions are weighted uniformly within each response, and each task record produces one optimizer update.

The key design choice is that supervision is evaluated on prefixes the student actually generates. This focuses feedback on learner-visited states (Bengio et al., 2015; Ranzato et al., 2016; Lin et al., 2020), while teacher-top-100 normalization focuses the objective on relative preferences among teacher-supported alternatives. No solution target or task reward enters the main objective; functional correctness is assessed only at evaluation by executing generated programs.

3.3 Fresh on-policy collection followed by fixed replay

Fig. 2(c) shows the two response schedules. In the first stage, the current image-conditioned student generates a fresh response for every update, and the text-view teacher provides distribution feedback along that response. As the student changes, the training prefixes therefore change with it.

The second stage trades this adaptive coverage for a stable coding distribution. The first-stage endpoint generates a response bank on the coding-task pool, and subsequent updates replay those stored responses with the same teacher and objective. Each complete run collects its own bank. The two stages therefore serve complementary roles: fresh sampling tracks the student's current policy, while fixed replay repeatedly fits a task-focused set of coding prefixes.

The paired-view design also supports controlled interventions on teacher information. For the teacher-view comparison, image-student conditions share initialization, response bank, update order, and optimizer settings while changing only the teacher's input view. For the solution-presence comparison, these components remain fixed while a ground-truth solution is appended only to the teacher context. In all cases, the student still receives the rendered task and learns only through distribution feedback. Appendix A.2 specifies the complete comparison families, including the separate text-student response bank.

4 Projected View Repair and Cross-View Transfer

Cross-modal disagreement is useful only when it points in a direction the student can realize and that realization improves the task. We formalize these two requirements, *i.e.*, *expressibility* and *task alignment*, at a fixed student checkpoint. Let $s = (x, u, y_{<t})$ denote a paired task and response prefix. On the teacher-selected support, define the text–image logit residual

$$r_s = z_s^T - z_s^I(0),$$

and let J_s^I be the Jacobian of the image-conditioned logits with respect to the trainable LoRA factors. Constant logit shifts are identified throughout. The distribution μ first samples a task and response and then samples uniformly among the retained response-token positions, matching Equation 4. Define

$$F_s = \mathrm{diag}(q_s^T) - q_s^T (q_s^T)^\top, \quad b_s = F_s^{1/2} r_s, \quad (\mathcal{B}_\mu \delta)_s = F_s^{1/2} J_s^I \delta. \tag{5}$$

Write

$$\mathcal{R}_\mu(\delta) = \mathbb{E}_\mu \mathrm{JSD}(q_s^I(\delta), q_s^T),$$

with $\|f\|_\mu^2 = \mathbb{E}_\mu \|f_s\|_2^2$, and let P_μ denote the orthogonal projection onto the range of $\mathcal{B}_\mu$.

Theorem 4.1 (Projected view repair)**.** *Fix the teacher support and prefix weights. Assume uniformly interior teacher probabilities and bounded student logit derivatives in a neighborhood where the centered residual and adapter displacement are $O(\varepsilon)$. Then*

$$\mathcal{R}_\mu(\delta) = \underbrace{\tfrac{1}{8}\|\mathcal{B}_\mu \delta - b\|_\mu^2}_{\mathcal{Q}_\mu(\delta)} + O(\varepsilon^3), \qquad \mathcal{Q}_\mu(0) - \min_\delta \mathcal{Q}_\mu(\delta) = \tfrac{1}{8}\|P_\mu b\|_\mu^2. \tag{6}$$

The minimizing displacement realizes this local prediction whenever it remains in the stated neighborhood.

The theorem gives the central geometric interpretation. Around the teacher distribution, JSD is locally $\frac{1}{8}v^\top F_s v + O(\|v\|^3)$ in the centered logit displacement v. Substituting the student's local expansion yields the quadratic objective $\mathcal{Q}_\mu$, whose least-squares solution projects the Fisher-weighted residual b onto the student's attainable tangent. Hence, raw cross-modal disagreement decomposes into a *repairable* component $P_\mu b$ and an orthogonal component that the current adapter cannot express.

This distinction can be summarized by the repairability ratio

$$\rho_\mu = \frac{\|P_\mu b\|_\mu^2}{\|b\|_\mu^2} \in [0, 1],$$

for $b \neq 0$. A shared correction satisfying $r_s = J_s^I \delta_\star + c_s \mathbf{1}$ gives $\rho_\mu = 1$. Importantly, this is a statement about the *local LoRA tangent*, not merely adapter rank. For a LoRA block at (B_0, A_0), the attainable first-order matrix perturbations are

$$\frac{\alpha}{r}(\Delta B A_0 + B_0 \Delta A).$$

Appendix B derives the expansion, retains the network-curvature remainder, and gives the corresponding shared-correction condition.

Useful teacher context. More teacher information is not necessarily more useful supervision. The text teacher induces a residual b, but only its projected component $P_\mu b$ is locally realizable by the student. If additional teacher context changes the residual by d, then under a common support, Fisher metric, and tangent,

$$U(b + d) - U(b) = 2\langle P_\mu b, P_\mu d\rangle_\mu + \|P_\mu d\|_\mu^2, \qquad U(b) = \|P_\mu b\|_\mu^2. \tag{7}$$

A context change can therefore increase total teacher–student disagreement while decreasing its useful projected component. Constructive alignment strengthens the repairable signal, whereas opposing alignment can weaken it. Each actual teacher context induces its own support and Fisher geometry; Equation 7 isolates the role of residual direction within a common reference geometry. The resulting principle is that teacher privilege matters through the correction the student can express, not through disagreement magnitude alone.

Task alignment and cross-view transfer. Expressibility is necessary but not sufficient: an attainable correction must also improve the evaluation task. Let d be the realized optimizer direction and G_V a smooth task utility for evaluation view V. For a step $\delta^+ = \eta d$,

$$G_V(\eta d) - G_V(0) = \eta \underbrace{\langle \nabla G_V(0), d\rangle}_{\Gamma_{V \leftarrow I}} + O(\eta^2 \|d\|^2). \tag{8}$$

Positive $\Gamma_{V \leftarrow I}$ implies improvement for sufficiently small steps. This formulation applies directly to the realized AdamW direction, including preconditioning, momentum, and weight decay.

For the quadratic-gradient case,

$$d = M g_I / 4, \qquad g_I = \mathbb{E}_\mu[(J_s^I)^\top F_s r_s],$$

with a fixed positive-semidefinite preconditioner M. Text-view transfer then depends on the cross-view kernel

$$J_s^T M (J_{s'}^I)^\top.$$

Because the LoRA parameters lie in language layers shared by both input routes, an update induced by image-conditioned training can also alter text-conditioned predictions. Whether that change helps depends on its alignment with the text-task utility.

From token corrections to task outcomes. The previous quantities describe continuous prediction and utility changes, whereas our evaluations use discrete exact-match and program-execution outcomes. To connect them, choose a reference response $y^\star$ whose postprocessed output either matches the transcription target or passes the coding tests, including its termination decision within the decoding budget. Let $m_{t,v}$ be the full-vocabulary logit margin of the desired token $y_t^\star$ over an alternative v at the reference prefix. If the margin Hessians have norm at most C along the update, then

$$\min_{t,\, v \neq y_t^\star} \left[m_{t,v}(0) + \eta \nabla m_{t,v}(0)^\top d \right] > \tfrac{1}{2} C \eta^2 \|d\|^2 \tag{9}$$

guarantees that greedy decoding follows $y^\star$.

The condition follows from the Taylor remainder: every desired token retains the largest logit, and induction yields the complete response. Thus, a continuous correction can change a discrete outcome by increasing a small number of bottleneck margins. This explains how modest distributional changes can convert an almost-correct transcription into an exact match, or move a generated program into the accepted set. Coding admits multiple valid programs, so each accepted response supplies a sufficient certificate of this form. Appendix B.6 develops the margin argument in detail.

Fresh sampling and fixed replay. The same geometry also clarifies the two training stages. Fresh sampling continually updates μ using the student's current responses, so supervision follows the prefixes the current policy visits. Fixed replay instead holds one empirical prefix distribution approximately fixed, enabling repeated fitting and matched teacher comparisons. For a fixed local model, changing prefix weights from μ to ν changes the quadratic-gradient direction by

$$g_\mu - g_\nu = \mathbb{E}_\mu[J^\top F r] - \mathbb{E}_\nu[J^\top F r].$$

The two stages therefore optimize the same distribution-feedback objective under different prefix weightings: fresh sampling provides adaptive coverage, while fixed replay provides a stable task-focused training distribution.

5 Experimental Design

We use Qwen3.5-4B (Qwen Team, 2026) with rank-64 language-layer LoRA and AdamW at 10^{-5}. Each of three complete self-OPD runs (seeds 17–19) performs 450 fresh-response updates on 256 transcription, continuation, and bug-fix instruction pairs, followed by 360 fixed-bank updates on coding tasks. Training inputs are rendered at 12 px while preserving source content. Ground-truth supervised fine-tuning (GT SFT) uses the same pretrained model, adapter, image inputs, and stage lengths, but trains on labelled transcription and solution targets as a separate recipe. Teacher-view and teacher-context studies use matched comparisons within their respective families. Appendix A gives prompts, data composition, hyperparameters, and the lexical-overlap check.

We evaluate coding on HumanEval (164 tasks) (Chen et al., 2021) and sanitized MBPP (427) (Austin et al., 2021), for 591 tasks in total. MBPP+ applies EvalPlus base and extra tests to 378 supported MBPP tasks using the same saved generations (Liu et al., 2023). Decoding is greedy with a 2048-token budget and thinking disabled, and all generation or test failures count as failures. We report pass@1, run means, sample SD, and two-sided exact McNemar tests for paired coding comparisons (McNemar, 1947; Dietterich, 1998). To measure cross-view transfer, we also evaluate the final image-trained checkpoints on text-input coding and exact transcription of 40 held-out snippets rendered at 12 px and 10 px. Appendix A.3 gives the complete grading protocol.

6 Results and Analysis

The results test the central prediction of projected view repair: useful cross-modal supervision depends not on disagreement alone, but on whether the induced correction is expressible, task-aligned, and shared across the model's input routes.

Observation 1: The largest gains occur where the initial text–image gap is largest, and survive stronger grading. Table 1 shows that mean image-input accuracy rises from 55.2% to 60.9%, outperforming both the pretrained model and GT SFT across all three runs. The gain concentrates on MBPP, where the pretrained text view solves 110 more tasks than the image view, compared with only 8 on HumanEval. This pattern is consistent with view repair: a larger representation gap creates more opportunity for an expressible, task-aligned correction. The improvement also survives stricter grading, with MBPP+ increasing from 46.6% to 52.6% on the same saved generations (Appendix C.3).

Observation 2: The stronger input view matters even when the teacher weights are identical. The matched comparison isolates input representation from model capacity (Table 2). From the same image-student start, image-view feedback ends at 339 solved tasks, whereas text-view feedback reaches 354, despite identical pretrained teacher weights. Their mean training JSDs are 0.89×10^{-3} and 2.81×10^{-3}, respectively (Fig. 3). Thus, a stronger representation of the same task can provide

Table 1: Image-input coding performance. All combines HumanEval and MBPP; stars indicate exact McNemar significance against the pretrained model.

Method	**HE**	**MBPP**	**MBPP+**	**All**
Pretrained	70.1	49.4	46.6	55.2
GT SFT (seed 17)	65.2	53.2	48.4	56.5
Δ vs. pretrained	−4.9	+3.7	+1.9	+1.4
Self-OPD (seed 17)	70.1	56.4**	52.1*	60.2*
Self-OPD (seed 18)	70.7	56.9***	52.4**	60.7**
Self-OPD (seed 19)	70.7	58.1***	53.2**	61.6***
Mean $\pm$ SD	70.5 $\pm$ 0.4	57.1 $\pm$ 0.8	52.6 $\pm$ 0.6	60.9 $\pm$ 0.7
Δ vs. pretrained	+0.4	+7.7	+6.0	+5.7
Δ vs. GT SFT	+5.3	+4.0	+4.1	+4.3

Table 2: Teacher-view controls under matched training. View pairs denote student / teacher; JSD is mean training divergence.

Student / teacher	**HE**	**MBPP**	**All**	**Δ All**	p	**JSD**
Start (image)	64.0	55.5	57.9	–	–	–
Image / image	66.5	53.9	57.4	−0.5	0.80	0.89
Image / text	71.3	55.5	59.9	+2.0	0.13	2.81
Start (text)	67.1	75.9	73.4	–	–	–
Text / text	76.2	75.4	75.6	+2.2	0.092	0.65

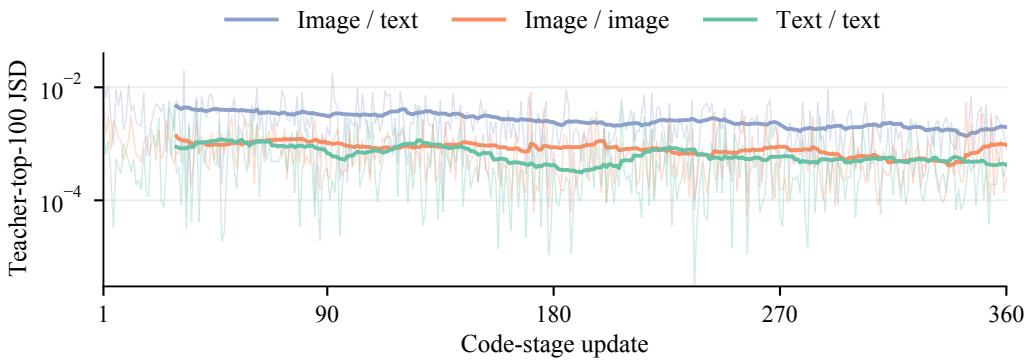

Figure 3. Teacher-top-100 JSD over all 360 code-stage updates. Legend pairs denote student / teacher input. Faint lines show raw values; solid lines show the trailing-30 mean on a logarithmic axis. Coding accuracy is measured on held-out tasks.

useful supervision without a stronger teacher model. Theorem 4.1 further shows why disagreement magnitude alone is insufficient: only its expressible, task-aligned component matters.

Observation 3: Repairing the weaker image route can also improve the stronger text route. The same image-trained checkpoints also improve under text input, raising mean coding accuracy from 75.1% to 76.4% (Table 7; Fig. 4). This rules out a purely input-specific interpretation of the gains. Because LoRA updates shared language layers, an image-induced correction can alter both input routes; Section 4 characterizes when that shared update is aligned with text-task utility.

Observation 4: The learned repair extends beyond the training renderer and converts near-correct reading into exact transcription. Exact transcription rises from 60.0% to 80.0% at 12 px and from 55.0% to 81.7% at the unseen 10 px density (Table 3). Because character similarity is already above 0.99, the gain is not simply coarse OCR improvement. Instead, correcting a few bottleneck tokens can convert an almost-correct sequence into an exact one, consistent with the margin analysis in Section 4.

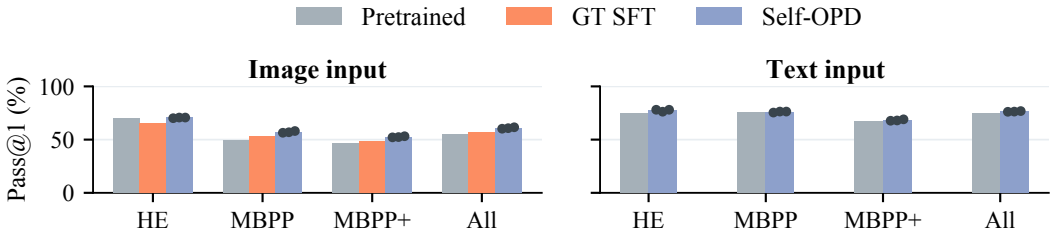

Figure 4. Image- and text-input coding performance of the pretrained model and image-trained checkpoints. HE denotes HumanEval; All combines HumanEval and MBPP.

Table 3: Exact transcription on 40 held-out snippets per rendering size. Self-OPD reports the mean and sample SD of three final coding checkpoints; improvements are relative to the pretrained model.

Model	**12 px: exact**	**12 px (%)**	**10 px: exact**	**10 px (%)**
Pretrained	24	60.0	22	55.0
Self-OPD (mean $\pm$ SD)	32.0 ± 1.7	80.0 ± 4.3	32.7 ± 1.5	81.7 ± 3.8
Δ vs. pretrained	+8.0	+20.0	+10.7	+26.7

Table 4: Teacher solution context (image-input pass@1, %). The blocks are separate matched studies, each with its own reference for Δ All and the paired p-value. Solutions enter only the teacher context; the student learns through distribution feedback.

Teacher context	**HE**	**MBPP**	**All**	**Δ All**	p
Task text only (reference)	70.1	56.4	60.2	–	–
Text + solution	68.9	54.1	58.2	-2.0	0.10
Full solution (reference)	71.3	55.5	59.9	–	–
Solution without docstrings	69.5	55.0	59.1	-0.8	0.52
Full solution + summary	67.1	55.3	58.5	-1.4	0.22

Observation 5: More teacher information does not guarantee more useful supervision. Appending a ground-truth solution to the teacher context reduces the matched seed-17 endpoint from 60.2% to 58.2% ($p = 0.10$; Table 4). A separate format study shows sensitivity to teacher context. Equation 7 explains the distinction: context changes not only disagreement magnitude but also its direction relative to the student's trainable tangent. Useful supervision therefore depends on the induced correction and its task alignment, not simply on how much information the teacher receives.

Observation 6: The training signal remains useful as learning shifts from adaptive prefix coverage to stable replay. The seed-17 trajectory rises from 326 to 352 solved tasks within the first 150 updates and ends at 356 after replay (Fig. 5). The stage transition has a direct interpretation: fresh collection tracks prefixes visited by the current student, whereas fixed replay holds the prefix distribution stable for repeated fitting. The two stages therefore serve complementary roles under the same distribution-matching objective.

7 Conclusion

A model can possess the knowledge needed for a task while accessing it unevenly across input representations. Cross-modal self-OPD turns this asymmetry into supervision: a stronger text view guides an image-conditioned student along the student's own response prefixes, without requiring a stronger external teacher, task rewards, or solution targets. Our theory of projected view repair shows that cross-modal disagreement is useful only to the extent that it is expressible by the student's adaptation tangent and aligned with task improvement, while shared language layers provide a route for transfer across views. Empirically, this mechanism improves image-input coding, exact transcription, and even text-input coding beyond the frozen teacher. Together, these results suggest that paired representations are not merely alternative interfaces to the same knowledge; they can provide a source of self-supervision for improving how that knowledge is accessed and used across modalities.

AI USAGE STATEMENT

Generative AI tools are used for language polishing and code assistance during manuscript preparation. The authors design the methods and experiments, interpret the results, verify the reported analyses, and make all final decisions regarding the scientific content. The authors retain full responsibility for the final manuscript, results, and accompanying code.

REPRODUCIBILITY STATEMENT

Section 3 specifies the training objective and response-collection procedure, while Section 5 describes training and evaluation. Appendix A provides prompts, rendering details, run configurations, and overlap checks; Appendix C reports per-run results, paired tests, and complete trajectories.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations (ICLR)*, 2024.

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Jackson Kernion, Kamal Ndousse, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, and Jared Kaplan. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Jiale Cheng, Yusen Liu, Xinyu Zhang, Yulin Fei, Wenyi Hong, Ruiliang Lyu, Weihan Wang, Zhe Su, Xiaotao Gu, Xiao Liu, Yushi Bai, Jie Tang, Hongning Wang, and Minlie Huang. Glyph: Scaling context windows via visual-text compression. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026.

Thomas G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1923, 1998. doi: 10.1162/089976698300017197.

Cheng Fan, Junyi Zhou, Tingzhang Luo, RongJian Xu, Qiyanhui Lu, Mingjian Zhu, Hanting Chen, and Jianyuan Guo. Reading is not reasoning: Bridging the agentic policy gap in vision-text compression. *arXiv preprint arXiv:2608.08960*, 2026.

Deqing Fu, Ruohao Guo, Ghazal Khalighinejad, Ollie Liu, Bhuwan Dhingra, Dani Yogatama, Robin Jia, and Willie Neiswanger. Isobench: Benchmarking multimodal foundation models on isomorphic representations. In *Conference on Language Modeling (COLM)*, 2024.

John Gkountouras, Josip Jukic, and Ivan Titov. Consensus as privileged context for label-free self-distillation. *arXiv preprint arXiv:2607.13643*, 2026.

Saurabh Gupta, Judy Hoffman, and Jitendra Malik. Cross modal distillation for supervision transfer. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

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 (ICLR)*, 2022.

Jonas Hubotter, Frederike Lubeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, and Andreas Krause. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*, 2026.

Geewook Kim, Teakgyu Hong, Moonbin Yim, Jeongyeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. Ocr-free document understanding transformer. In *European Conference on Computer Vision (ECCV)*, 2022.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2016.

Kenton Lee, Mandar Joshi, Iulia Turc, Hexiang Hu, Fangyu Liu, Julian Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. Pix2struct: Screenshot parsing as pretraining for visual language understanding. In *International Conference on Machine Learning (ICML)*, 2023.

Kaixin Li, Yuchen Tian, Qisheng Hu, Ziyang Luo, Zhiyong Huang, and Jing Ma. Mmcode: Benchmarking multimodal large language models for code generation with visually rich programming problems. In *Findings of the Association for Computational Linguistics: EMNLP*, 2024.

Pengyu Li, Zhitao Gao, Lingling Zhang, Muye Huang, Yuanming Li, Zesheng Yang, Fangzhi Xu, and Jun Liu. Visual-opsd: Cross-modal on-policy self-distillation for efficient unified multimodal reasoning. *arXiv preprint arXiv:2606.18974*, 2026a.

Yijiang Li, Yijun Liang, Yunjie Tian, Bingyang Wang, Ke Zhang, Zhenfei Yin, Di Fu, Philip Torr, and Nuno Vasconcelos. Self-supervised visual on-policy distillation. *arXiv preprint arXiv:2608.14144*, 2026b.

Alexander Lin, Jeremy Wohlwend, Howard Chen, and Tao Lei. Autoregressive knowledge distillation through imitation learning. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

J. Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1): 145–151, 1991. doi: 10.1109/18.61115.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

David Lopez-Paz, Leon Bottou, Bernhard Scholkopf, and Vladimir Vapnik. Unifying distillation and privileged information. In *International Conference on Learning Representations (ICLR)*, 2016.

Kevin Lu and Thinking Machines Lab. On-policy distillation. Thinking Machines Lab: Connectionism (blog post), https://thinkingmachines.ai/blog/on-policy-distillation/, 2025. URL https://thinkingmachines.ai/blog/on-policy-distillation/.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947. doi: 10.1007/bf02295996.

Qwen Team. Qwen3.5: Towards native multimodal agents. Blog post, https://qwen.ai/blog?id=qwen3.5, 2026. URL https://qwen.ai/blog?id=qwen3.5.

Pooyan Rahmanzadehgervi, Logan Bolton, Mohammad Reza Taesiri, and Anh Totti Nguyen. Vision language models are blind. In *Asian Conference on Computer Vision*, pp. 293–309. Springer, 2024.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *International Conference on Learning Representations (ICLR)*, 2016.

Stephane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635, 2011. URL https://proceedings.mlr.press/v15/ross11a.html.

Phillip Rust, Jonas F. Lotz, Emanuele Bugliarello, Elizabeth Salesky, Miryam de Lhoneux, and Desmond Elliott. Language modelling with pixels. In *International Conference on Learning Representations (ICLR)*, 2023.

Idan Shenfeld, Mehul Damani, Jonas Hubotter, and Pulkit Agrawal. Self-distillation enables continual learning. *arXiv preprint arXiv:2601.19897*, 2026.

Charlie Snell, Dan Klein, and Ruiqi Zhong. Learning by distilling context. *arXiv preprint arXiv:2209.15189*, 2022.

Alex Stein, Furong Huang, and Tom Goldstein. Gates: Self-distillation under privileged context with consensus gating. *arXiv preprint arXiv:2602.20574*, 2026.

Kaiser Sun, Xiaochuang Yuan, Hongjun Liu, Chen Zhao, Cheng Zhang, Mark Dredze, and Fan Bai. Reading, not thinking: Understanding and bridging the modality gap when text becomes pixels in multimodal llms. *arXiv preprint arXiv:2603.09095*, 2026.

Vladimir Vapnik and Akshay Vashist. A new learning paradigm: Learning using privileged information. *Neural Networks*, 22(5-6):544–557, 2009. doi: 10.1016/j.neunet.2009.06.042.

Haoran Wei, Yaofeng Sun, and Yukun Li. Deepseek-ocr: Contexts optical compression. *arXiv preprint arXiv:2510.18234*, 2025.

Chengyue Wu, Zhixuan Liang, Yixiao Ge, Qiushan Guo, Zeyu Lu, Jiahao Wang, Ying Shan, and Ping Luo. Plot2code: A comprehensive benchmark for evaluating multi-modal large language models in code generation from scientific plots. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 3006–3028, 2025.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

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

John Yang, Carlos E. Jimenez, Alex L. Zhang, Kilian Lieret, Joyce Yang, Xindi Wu, Ori Press, Niklas Muennighoff, Gabriel Synnaeve, Karthik R. Narasimhan, Diyi Yang, Sida I. Wang, and Ofir Press. Swe-bench multimodal: Do ai systems generalize to visual software domains? In *International Conference on Learning Representations (ICLR)*, 2025b.

Qianhao Yuan, Jie Lou, Xing Yu, Hongyu Lin, Le Sun, Xianpei Han, and Yaojie Lu. Vision-opd: Learning to see fine details for multimodal llms via on-policy self-distillation. *arXiv preprint arXiv:2605.18740*, 2026.

Sukmin Yun, Haokun Lin, Rusiru Thushara, Mohammad Qazim Bhat, Yongxin Wang, Zutao Jiang, Mingkai Deng, Jinhong Wang, Tianhua Tao, Junbo Li, Haonan Li, Preslav Nakov, Timothy Baldwin, Zhengzhong Liu, Eric P. Xing, Xiaodan Liang, and Zhiqiang Shen. Web2code: A large-scale webpage-to-code dataset and evaluation framework for multimodal llms. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2024.

Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, et al. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, pp. 169–186. Springer, 2024.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

A Experimental Design Details

A.1 Prompt construction and rendering

Views. Both views contain the same underlying task content and task instruction, with the prompt layouts specified below. In the image view the user message contains the rendered task pages followed by the instruction. In the text view used by the teacher during training, the user message contains the

Table 5: Training constants shared by all self-OPD runs and, where applicable, by GT SFT.

Setting	Value
Model / snapshot	Qwen3.5-4B / 851bf6e8
LoRA rank / scaling	64 / 128
LoRA dropout / bias	0 / none
Adapted projections	q, k, v, o, gate, up, down (language model only)
Base / loss precision	bfloat16 / float32 for selected logits
Gradient checkpointing	Enabled
Generation cache	Disabled during training
Optimizer / gradient clip	AdamW / 1.0
Update unit	One task record
Self-OPD learning rate	10^{-5} in both stages
GT SFT learning rates	5×10^{-5} (transcription), 10^{-5} (solution)
Sampling	Temperature 0.7, top-p 0.9
Generated-token limits	192 (first stage), 512 (second-stage collection)
Loss support	Teacher top 100; separate renormalization
Loss mixture / weighting	Generalized JSD, $\beta = 0.5$ / uniform positions
Evaluation decoding	Greedy, up to 2048 new tokens, thinking disabled

instruction, a code-as-text lead-in sentence and a Python code fence enclosing the source content, and the phrases "shown in the image" and "in the image" in the instruction are replaced by "below". In the text-input evaluation and in the text-student control, the raw task text is supplied without a lead-in sentence or code fence. The chat template adds the assistant generation prefix, and the same student-generated response ids follow each prompt when the aligned student and teacher logits are computed.

Instructions. First-stage transcription requests exact code copying; continuation requests a plausible next function or method in the same domain and style; bug fixing requests a one-sentence purpose description followed by corrected code. The coding evaluation uses the single instruction "solve the problem shown in the image; output only the code" for both views. The transcription evaluation uses the instruction "Transcribe the code shown in this image exactly, preserving all whitespace, indentation and comments. Output only the code, no commentary."

Rendering. The coding renderer typesets monospaced black text in DejaVu Sans Mono at 12 px with a 16 px line height and 8 px padding on a white background. The page width is fitted to the widest line up to a 100-character wrap ceiling and clamped to 320–1280 px; hard wrapping and page breaks preserve the full content, with a maximum page height of 2200 px before final padding, and each page's dimensions are padded upward to multiples of 32 px. The first-stage renders use the same geometry. The transcription evaluation uses two separate geometries: 12 px font, 15 px line height, 12 px padding, 724 px width; and 10 px font, 13 px line height, 12 px padding, 608 px width.

A.2 Runs and comparison families

Main runs. Seeds 17, 18 and 19 each train a first stage from the pretrained model with a fresh adapter (450 updates, fresh samples of at most 192 tokens, temperature 0.7, nucleus 0.9), collect a code-stage response bank of 360 responses of at most 512 tokens with that seed's own first-stage endpoint, and train that seed's code stage (360 updates) by replaying the bank. The frozen pretrained model is the teacher throughout. Table 5 lists the shared hyperparameters.

GT SFT run. Ground-truth supervised fine-tuning uses the same pretrained checkpoint, adapter configuration, image-conditioned inputs, and stage counts. It performs 450 cross-entropy updates at 5×10^{-5} on a 300-example transcription pool, then 360 updates at 10^{-5} on supplied solutions, initialized from the transcription endpoint; prompt tokens are masked. All 256 first-stage self-OPD source IDs belong to the 300-example transcription split. GT SFT was trained once with seed 17. Because its first-stage objective, instruction pool, and learning rate differ, the comparison evaluates complete recipes.

Teacher-view family (Table 2). All three conditions initialize from the seed-17 first-stage endpoint and run 360 code-stage updates with the frozen pretrained teacher, no ground-truth solution, uniform weights and the main optimizer settings, using torch seed 19 and record-shuffle seed 19. The two image-student conditions replay the same image-student response bank in the same order and differ

only in the teacher's input view; the text-student condition uses its own text-student response bank and is compared with the shared start evaluated on text. This matched comparison continues from the shared first-stage checkpoint. The three complete main runs each use their own first-stage training and collected responses.

Teacher-solution family (Table 4, top). The condition without a solution is the seed-17 main endpoint; the condition with a solution reruns the same 360 updates on the same response bank with the ground-truth solution appended to the teacher's problem text inside reference-solution tags. The student never receives the solution and never receives a cross-entropy target.

Solution-format family (Table 4, bottom). A separate matched family (torch seed 17, order seed 19, 360 updates per condition) compares the full ground-truth solution, the ground-truth solution with standalone function docstrings removed (preserving the remaining bytes and the executable abstract syntax tree), and the full ground-truth solution followed by a separate reviewed solution summary generated with the pretrained model. Student input, replayed responses, update order, initialization, optimizer and objective are held fixed; comparisons are within the family, against the full-reference condition.

A.3 Task pools and evaluation protocol

The first-stage pool contains 256 instruction pairs from a fixed repository code-snippet corpus: 86 transcription, 85 continuation, and 85 bug-fix instructions. The second-stage pool contains 360 single-round coding tasks: 140 bug fixes, 120 code completions, and 100 terminal diagnoses with captured tracebacks. Each record contains a code module, instruction, and supplied solution; the main method uses the module and instruction as task inputs.

HumanEval provides its function signature and specification; sanitized MBPP provides the description followed by its visible tests. The image-input evaluation renders that problem text, and the text-input evaluation supplies it directly. Greedy decoding permits at most 2048 new tokens with thinking disabled. Every task is retained, including generation and test failures, and pass@1 is $100 \times$ solved/graded. Combined pools all 591 HumanEval and MBPP tasks. MBPP+ requires EvalPlus base and extra tests on the same saved generations for 378 supported MBPP tasks (Appendix C.3).

Transcription uses 40 code snippets held out from training, evaluated under each of the two rendering geometries. We extract the first fenced code block when present, otherwise the output text, strip trailing whitespace on each line, and remove trailing empty lines. Internal indentation and line breaks are preserved. Exact match compares the resulting complete strings. Each final image-trained checkpoint is evaluated on both text-input coding and transcription.

The three main runs repeat the entire training pipeline and collect their own responses. We report sample SD across runs and compute percentage-point differences from unrounded counts. Two-sided exact McNemar tests condition on the discordant task pairs; all per-run counts and paired comparisons are retained in Appendix C.

A.4 Training–benchmark overlap check

For each training pool (first-stage instruction plus code text; code-stage module text plus instruction plus reference; GT-SFT transcription source text) and each benchmark text (HumanEval prompt plus canonical solution; MBPP prompt plus code), lower-cased ASCII word tokens matching `[A-Za-z0-9_]+` were compared in contiguous windows of 20 tokens, retaining windows with at least seven alphabetic tokens. Evaluation documents without an eligible window and with a normalized length of at least eight characters were additionally tested as NFKC- and whitespace-normalized lower-cased substrings of the normalized training text. No window matched and no short-document fallback hit occurred in any of the six pool–benchmark combinations. Including reference code extends the check beyond problem wording, while token-based matching makes the window comparison insensitive to differences in whitespace.

B Projected View Repair: Derivations and Consequences

B.1 Local geometry and uniform assumptions

Fix a student checkpoint, a frozen teacher, and a prefix distribution μ. A draw from μ selects a task and a generated response according to the current collection scheme and then a retained response position uniformly. This ordering gives each task response the uniform-position loss in Equation 4.

The empirical replay distribution uses its stored task records and prefixes with these same weights. Each state has a fixed teacher-selected support K_s, of size at most 100. Work on the logit quotient obtained by centering each support vector with $C_s = I - \mathbf{1}\mathbf{1}^\top/|K_s|$.

Assume the selected teacher probabilities satisfy $q_s^T(v) \geq a > 0$ uniformly, and that the centered student logit map has uniformly bounded first and second derivatives in the neighborhood considered. Write

$$C_s\{z_s^I(\delta) - z_s^T\} = e_s(\delta) + h_s(\delta), \qquad e_s(\delta) = J_s^I\delta - r_s, \qquad \|h_s(\delta)\| \leq C\|\delta\|^2, \tag{10}$$

where r_s and J_s^I are centered representatives. Uniform bounds make the pointwise remainder integrable under μ. In the joint small-perturbation regime, $\operatorname{ess\,sup}_s \|r_s\| = O(\varepsilon)$ and $\|\delta\| = O(\varepsilon)$. The teacher support, its probabilities, and the sampling weights remain fixed while this local update is analyzed.

For an interior distribution q and a probability perturbation u with $\mathbf{1}^\top u = 0$, Taylor expansion yields

$$\mathrm{JSD}(q+u, q) = \tfrac{1}{8}u^\top \operatorname{diag}(q)^{-1}u + O(\|u\|^3). \tag{11}$$

Let $q = \mathrm{softmax}(z)$ and $F = \operatorname{diag}(q) - qq^\top$. Softmax differentiation gives $u = Fv + O(\|v\|^2)$, while $F\operatorname{diag}(q)^{-1}F = F$. Hence

$$\mathrm{JSD}(\mathrm{softmax}(z+v), \mathrm{softmax}(z)) = \tfrac{1}{8}v^\top Fv + O(\|v\|^3). \tag{12}$$

The identity $F\mathbf{1} = 0$ makes this expansion independent of constant logit shifts. Its assumptions concern the centered logit perturbations and the local derivative bounds, which specify the geometry used below.

B.2 Projection theorem and the nonlinear remainder

Substitution of Equation 10 into Equation 12, followed by averaging, gives the more explicit remainder bound

$$\mathcal{R}_\mu(\delta) = \tfrac{1}{8}\|\mathcal{B}_\mu\delta - b\|_\mu^2 + E_\mu(\delta), \tag{13}$$

$$|E_\mu(\delta)| \leq C'\left(\mathbb{E}_\mu\|e_s(\delta)\|^3 + \|\delta\|^2\mathbb{E}_\mu\|e_s(\delta)\| + \|\delta\|^4\right). \tag{14}$$

The cross product with network curvature gives the second term, and squared curvature gives the last. The joint regime yields $O(\varepsilon^3)$, proving Theorem 4.1; exact linearized cancellation can leave fourth-order risk.

The adapter parameter space is finite dimensional, so the bounded operator $\mathcal{B}_\mu$ has a closed finite-dimensional range in the centered $L^2(\mu)$ space. Its orthogonal projection therefore exists. Decomposing

$$b = P_\mu b + (I - P_\mu)b \tag{15}$$

and applying the Pythagorean identity gives

$$\min_\delta \mathcal{Q}_\mu(\delta) = \tfrac{1}{8}\|(I-P_\mu)b\|_\mu^2, \qquad \mathcal{Q}_\mu(0) - \min_\delta \mathcal{Q}_\mu(\delta) = \tfrac{1}{8}\|P_\mu b\|_\mu^2. \tag{16}$$

This is an exact identity for the quadratic model. With an explicit displacement budget $\|\delta\| \leq R$, the same value is attained whenever a least-squares minimizer lies in that ball. The nonlinear risk then follows the remainder bound above. The minimum-norm minimizer $\mathcal{B}_\mu^\dagger b$ gives a direct way to check this condition.

B.3 Shared corrections in the actual LoRA tangent

For an adapted block $W = W_0 + (\alpha/r)BA$ at factors (B_0, A_0), differentiation gives

$$\Delta W_{\tan} = \frac{\alpha}{r}(\Delta B A_0 + B_0 \Delta A). \tag{17}$$

Let h_s be the block's input and D_s the downstream Jacobian from the block output to the selected logits. A blockwise first-order logit correction is $D_s\Delta W_{\tan}h_s$; contributions from multiple adapted blocks add. Suppose a common factor displacement produces

$$r_s = \sum_\ell D_{s,\ell}\frac{\alpha_\ell}{r_\ell}(\Delta B_\ell A_{0,\ell} + B_{0,\ell}\Delta A_\ell)h_{s,\ell} + c_s\mathbf{1} \quad \mu\text{-almost surely.} \tag{18}$$

Then $b = \mathcal{B}_\mu\delta_\star$, and $\rho_\mu = 1$ for $b \neq 0$. Approximate shared corrections are quantified by their residual $(I - P_\mu)b$.

Equation 17 specifies local attainability at the current factors. For example, at $B_0 = 0$ with full-row-rank A_0, every matrix correction whose row space lies in the row space of A_0 can be written as $\Delta B A_0$. A general rank-r matrix has a global factorization BA; its local realization at a particular checkpoint is governed by Equation 17. The shared-correction condition therefore refers to the current trained parameterization.

B.4 The contribution of additional teacher context

Under a common support, reference metric, student tangent, and prefix weighting, linearity of $P = P_\mu$ gives

$$\|P(b+d)\|^2 - \|Pb\|^2 = 2\langle Pb, Pd\rangle + \|Pd\|^2, \tag{19}$$

which proves Equation 7. It separates constructive, opposing, and orthogonal contributions from the context shift. To exhibit both a larger total residual and a smaller projected one within the small-perturbation regime, take unit vectors $v \in \mathrm{range}(P)$ and $u \in \ker(P)$ in the centered logit-feature space. Such vectors exist when the range is a nonzero proper subspace. Set

$$b = \varepsilon v, \qquad d = \varepsilon(2u - v). \tag{20}$$

Then $\|b+d\|^2 = 4\varepsilon^2$ versus $\|b\|^2 = \varepsilon^2$, while $\|P(b+d)\|^2 = 0$ versus $\|Pb\|^2 = \varepsilon^2$. Both residuals remain $O(\varepsilon)$. Thus context value is determined by projection and alignment even at arbitrarily small scales.

For actual text, image, and solution-conditioned teachers, support selection and F_s are teacher dependent. Each context is evaluated with its own $(\mathcal{B}_V, b_V, P_V)$; Equation 7 isolates residual rotation in a common reference geometry. The matched experiments compare complete teacher conditions, each in its own geometry.

B.5 Task utility, the optimizer, and cross-view transfer

Let G_V have Hessian norm bounded by L on the segment from 0 to ηd. Taylor's theorem gives

$$|G_V(\eta d) - G_V(0) - \eta\langle \nabla G_V(0), d\rangle| \le \tfrac{1}{2} L\eta^2 \|d\|^2. \tag{21}$$

If $\Gamma_{V\leftarrow I} > 0$, the utility increases for $0 < \eta < 2\Gamma_{V\leftarrow I}/(L\|d\|^2)$, with the usual interpretation when $L = 0$. This statement takes the realized update direction as its argument. In AdamW, that direction includes the bias-corrected first moment, the diagonal normalization from the second moment, and weight decay. It therefore incorporates the optimizer's state as well as the current gradient.

For the quadratic risk,

$$\nabla \mathcal{Q}_\mu(0) = -\tfrac{1}{4} g_I, \qquad g_I = \mathbb{E}_\mu[(J_s^I)^\top F_s r_s]. \tag{22}$$

In the preconditioned-gradient case $d = M g_I/4$, let ν denote the text evaluation distribution and let a_s^T be a smooth text utility's gradient with respect to its logits. The chain rule yields

$$\Gamma_{T\leftarrow I} = \tfrac{1}{4}\mathbb{E}_{s\sim\nu,\, s'\sim\mu}\left[(a_s^T)^\top J_s^T M (J_{s'}^I)^\top F_{s'} r_{s'}\right]. \tag{23}$$

The evaluation Jacobian can use the full output vocabulary while the training Jacobian uses its selected support. Shared language-layer parameters couple the two views through this kernel. Its signed contraction with the task and teacher residual directions determines the first-order utility change. For a realized AdamW step, Equation 21 applies directly to its direction d.

B.6 Exact transcription and executable decision margins

Choose a reference response $y^\star$ within the decoding budget and include its termination decision. For transcription it is the normalized target snippet; for coding choose an accepted program. At every reference prefix define the full-vocabulary margins

$$m_{j,t,v}(\theta) = z_{j,t}(y_t^\star;\theta) - z_{j,t}(v;\theta), \qquad v \ne y_t^\star. \tag{24}$$

If all these margins are strictly positive, greedy decoding follows $y^\star$ by induction over positions, giving an exact transcription or an accepted program. This provides a sufficient certificate for the discrete task outcome. Code evaluation can accept multiple programs, so each accepted reference supplies such a certificate.

Suppose the margin Hessians along the update segment are bounded by C. A concrete sufficient condition after an update is

$$\min_{t, v\ne y_t^\star}\left\{m_{j,t,v}(0) + \eta \nabla m_{j,t,v}(0)^\top d\right\} > \tfrac{1}{2} C\eta^2\|d\|^2. \tag{25}$$

Equation 25 is the certificate in Section 4. Shared corrections can raise bottleneck margins across snippets; benchmark differences depend on their initial margin distributions and the update. The observed solved-count gaps provide outcome-level context for this conditional mechanism.

B.7 Prefix weighting in fresh and replay training

At a fixed local model, let $h_s = (J_s^I)^\top F_s r_s$ and assume $\|h_s\| \le H$. If total variation is defined by $\mathrm{TV}(\mu,\nu) = \frac{1}{2}\|\mu - \nu\|_1$, then

$$\|g_\mu - g_\nu\| = \left\| \int h_s \, d(\mu - \nu)(s) \right\| \le 2H \, \mathrm{TV}(\mu,\nu). \tag{26}$$

Fresh sampling changes prefix coverage as the student changes, while a stored response bank fixes these empirical weights. For a fixed bank and fixed quadratic model, put $H_\mu = \mathcal{B}_\mu^* \mathcal{B}_\mu$. Gradient descent satisfies

$$\delta_{k+1} = \delta_k + \tfrac{\eta}{4}(g_\mu - H_\mu \delta_k). \tag{27}$$

When $H_\mu \neq 0$ and $0 < \eta < 8/\lambda_{\max}(H_\mu)$, each active eigendirection contracts toward its least-squares value. This describes consolidation in the fixed local model. In the training implementation, refreshing prefixes changes their weights, and updating the parameters changes the local Jacobians; the stage design determines which of these sources of variation is present. Token IDs are held fixed during each differentiated update in both stages.

C Additional Results and Diagnostics

C.1 Paired tests for the main comparison

Table 6 lists, for every image-input comparison in Table 1, the solved counts of both models, the number of tasks solved by the treatment but not the comparator (gained), the reverse (lost), the net change and the two-sided exact McNemar p-value computed from the binomial distribution of the discordant pairs. The self-OPD runs beat the pretrained model on Combined, MBPP and MBPP+ in every run at $p < 0.02$ and beat the GT SFT run on Combined in every run at $p < 0.035$; HumanEval net differences from the pretrained model are 0, $+1$ and $+1$ tasks ($p = 1.00$ in every run).

C.2 Text-input results per seed

Table 8 gives the per-seed text-input results behind Table 7. All twelve per-seed differences from the pretrained text model are positive in solved count.

C.3 Three correctness criteria on the matched MBPP tasks

The 378 MBPP task ids supported by EvalPlus were graded three ways on the same saved generations: with the original MBPP tests, with the EvalPlus base tests, and with the MBPP+ criterion that requires both base and extra tests to pass. Table 9 shows improvements under all three criteria for image input and positive mean differences for text input. MBPP+ applies stricter grading to a subset of MBPP. Combined is computed over the evaluated HumanEval and MBPP sets.

C.4 Transcription details

Transcription was scored on 40 held-out rendered-code examples per rendering condition (Section 5). Table 10 reports the exact-match counts together with the mean character-level similarity, defined as the difflib sequence-matcher ratio between the normalized prediction and reference (twice the number of matching characters divided by the sum of the two lengths, without automatic junk heuristics). Character similarity measures normalized string similarity, while exact match requires equality of the complete normalized snippet.

C.5 Teacher solution context

The solution-presence and solution-format families each retain a matched reference (Table 4). In the first, the task-text reference is the seed-17 main endpoint. In the second, the full-solution condition supplies the reference for each format change. Appendix A.2 specifies their initialization, response collection, and ordering.

Table 6: Paired exact McNemar tests on image-input results (solved counts, gained/lost tasks, net, two-sided p). Each self-OPD seed is compared with the pretrained model and with the GT SFT run on the same tasks.

Comparison	Benchmark	Treat.	Comp.	Gained	Lost	Net	p
GT SFT vs. pretrained	HE	107	115	20	28	-8	0.31
	MBPP	227	211	50	34	+16	0.10
	MBPP+	183	176	37	30	+7	0.46
	All	334	326	70	62	+8	0.54
OPD (17) vs. pretrained	HE	115	115	20	20	+0	1.00
	MBPP	241	211	61	31	+30	0.002
	MBPP+	197	176	47	26	+21	0.019
	All	356	326	81	51	+30	0.011
OPD (18) vs. pretrained	HE	116	115	22	21	+1	1.00
	MBPP	243	211	60	28	+32	<0.001
	MBPP+	198	176	42	20	+22	0.007
	All	359	326	82	49	+33	0.005
OPD (19) vs. pretrained	HE	116	115	20	19	+1	1.00
	MBPP	248	211	63	26	+37	<0.001
	MBPP+	201	176	46	21	+25	0.003
	All	364	326	83	45	+38	<0.001
OPD (17) vs. GT SFT	HE	115	107	21	13	+8	0.23
	MBPP	241	227	39	25	+14	0.10
	MBPP+	197	183	34	20	+14	0.076
	All	356	334	60	38	+22	0.033
OPD (18) vs. GT SFT	HE	116	107	22	13	+9	0.18
	MBPP	243	227	42	26	+16	0.068
	MBPP+	198	183	35	20	+15	0.058
	All	359	334	64	39	+25	0.018
OPD (19) vs. GT SFT	HE	116	107	24	15	+9	0.20
	MBPP	248	227	46	25	+21	0.017
	MBPP+	201	183	36	18	+18	0.020
	All	364	334	70	40	+30	0.005

Table 7: Text-input pass@1 (%) for the final image-trained checkpoints. Self-OPD reports the mean and sample SD of three runs; the pretrained model supplied the frozen teacher. Evaluation sets follow Table 1.

Model	HE	MBPP	MBPP+	All
Pretrained	75.0	75.2	67.2	75.1
Self-OPD (mean $\pm$ SD)	77.4 ± 1.1	76.0 ± 0.5	68.3 ± 0.7	76.4 ± 0.4
Δ vs. pretrained	+2.4	+0.9	+1.1	+1.3

C.6 Learning-curve values

Table 11 lists the checkpoint values plotted in Fig. 5. Both recipes were evaluated at 0, 150, 300 and 450 updates of their first stage and at 120, 240 and 360 updates of their code stage, with the first-stage endpoint reused as the code-stage starting point. The last scheduled checkpoint is the reported endpoint of each recipe. Paired tests between consecutive checkpoints on Combined are: self-OPD first stage +4.4 pp ($p = 0.022$), -0.8 pp ($p = 0.51$), -0.8 pp ($p = 0.60$); self-OPD code stage +2.7 pp ($p = 0.052$), -0.7 pp ($p = 0.64$), +0.3 pp ($p = 0.88$); GT SFT transcription stage +2.0 pp ($p = 0.31$), -10.2 pp ($p < 0.001$), +5.2 pp ($p < 0.001$); GT SFT solution stage +5.6 pp ($p < 0.001$), -0.5 pp ($p = 0.69$), -0.8 pp ($p = 0.47$).

Table 8: Text-input results per seed (solved / graded, pass@1 %) with the exact McNemar test against the pretrained text model.

Model	Benchmark	Solved	%	Gained	Lost	p
Pretrained	HE	123 / 164	75.0	–	–	–
	MBPP	321 / 427	75.2	–	–	–
	MBPP+	254 / 378	67.2	–	–	–
	All	444 / 591	75.1	–	–	–
Self-OPD seed 17	HE	128 / 164	78.0	12	7	0.36
	MBPP	322 / 427	75.4	17	16	1.00
	MBPP+	256 / 378	67.7	14	12	0.85
	All	450 / 591	76.1	29	23	0.49
Self-OPD seed 18	HE	125 / 164	76.2	13	11	0.84
	MBPP	326 / 427	76.3	23	18	0.53
	MBPP+	257 / 378	68.0	18	15	0.73
	All	451 / 591	76.3	36	29	0.46
Self-OPD seed 19	HE	128 / 164	78.0	12	7	0.36
	MBPP	326 / 427	76.3	21	16	0.51
	MBPP+	261 / 378	69.0	18	11	0.26
	All	454 / 591	76.8	33	23	0.23

Table 9: Three correctness criteria applied to the same saved generations on the 378 MBPP tasks supported by EvalPlus (pass@1 %). MBPP+ requires both the EvalPlus base and extra tests to pass.

Input	Model	Original tests	EvalPlus base	MBPP+
Image	Pretrained	52.9	53.2	46.6
	GT SFT (seed 17)	57.4	57.1	48.4
	Self-OPD (seed 17)	60.3	60.8	52.1
	Self-OPD (seed 18)	61.1	61.9	52.4
	Self-OPD (seed 19)	61.6	62.4	53.2
	Mean ± SD	61.0 ± 0.7	61.7 ± 0.8	52.6 ± 0.6
	Δ vs. pretrained	+8.1	+8.6	+6.0
Text	Pretrained	78.0	78.6	67.2
	Self-OPD (seed 17)	78.0	79.1	67.7
	Self-OPD (seed 18)	80.2	81.0	68.0
	Self-OPD (seed 19)	79.4	80.7	69.0
	Mean ± SD	79.2 ± 1.1	80.2 ± 1.0	68.3 ± 0.7
	Δ vs. pretrained	+1.1	+1.7	+1.1

Table 10: Transcription details: exact matches and mean character similarity (difflib ratio on normalized strings) for each model and rendering size.

Model	12 px: exact	12 px: similarity	10 px: exact	10 px: similarity
Pretrained	24	0.9922	22	0.9926
Self-OPD seed 17	33	0.9956	33	0.9882
Self-OPD seed 18	33	0.9966	34	0.9987
Self-OPD seed 19	30	0.9966	31	0.9949

C.7 Divergence traces

Table 12 summarizes the raw per-update JSD traces plotted in Fig. 3. Each value is the training step's teacher-top-100 JSD at $\beta = 0.5$, averaged uniformly over the generated positions of that step's response; every one of the 360 updates per condition is included, and the stored raw values are reported without clipping or smoothing. The trailing mean at step k is the arithmetic mean of the raw values from $\max(1, k - 29)$ to k, computed independently within each condition. The divergence measures next-token distribution disagreement on the training responses; held-out coding outcomes are reported in Table 2.

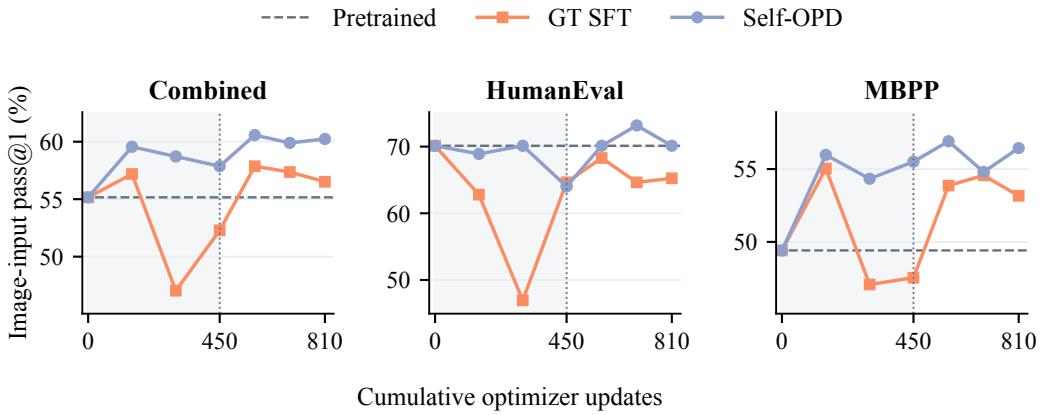

Figure 5. Image-input pass@1 at every saved checkpoint of the seed-17 runs. Panels show All (591 tasks), HumanEval (164), and MBPP (427). The marker at 450 updates separates stages; shading marks the first stage. Endpoints are the last scheduled checkpoints.

Table 11: Image-input pass@1 (%) at every saved checkpoint of the seed-17 runs (values plotted in Fig. 5). Step 0 of the first stage is the pretrained model; step 0 of the code stage is the first-stage endpoint.

Recipe	Stage	Step	HE	MBPP	All
Self-OPD	first	0	70.1	49.4	55.2
	first	150	68.9	56.0	59.6
	first	300	70.1	54.3	58.7
	first	450	64.0	55.5	57.9
	code	0	64.0	55.5	57.9
	code	120	70.1	56.9	60.6
	code	240	73.2	54.8	59.9
	code	360	70.1	56.4	60.2
GT SFT	first	0	70.1	49.4	55.2
	first	150	62.8	55.0	57.2
	first	300	47.0	47.1	47.0
	first	450	64.6	47.5	52.3
	code	0	64.6	47.5	52.3
	code	120	68.3	53.9	57.9
	code	240	64.6	54.6	57.4
	code	360	65.2	53.2	56.5

Table 12: Summaries of the raw per-update teacher-top-100 JSD traces ($\times 10^{-3}$; 360 updates per condition). Final MA is the trailing-30 mean at update 360; it equals the mean of the last 30 raw values.

Student / teacher	Mean	Median	Max	First 30	Last 30	Final MA
Image / text	2.81	2.14	19.52	4.66	1.98	1.98
Image / image	0.89	0.61	8.34	1.37	0.96	0.96
Text / text	0.65	0.37	5.92	0.88	0.43	0.43

Each step averages over its retained response positions, and trailing means average these step-level losses. The fixed bank holds prefixes constant as the student changes. Thus, longer responses do not receive greater weight in the step-averaged summaries.