SELF-BOOSTING VISION-LANGUAGE MODELS WITH NOISY STUDENT ON-POLICY SELF-DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-training enables vision-language models (VLMs) to understand human instructions and perform various downstream tasks. Current post-training methods usually rely on human-annotated data, distillation from external models, reinforcement learning with human feedback, or verifiable answers. This limits their ability to improve without external supervision. To tackle this, we propose NOPD (Noisy Student On-Policy Self-Distillation), a simple yet effective self-distillation approach that improves VLMs without any external models or ground-truth answers. Our key insight is that prediction discrepancies between clean and corrupted inputs naturally induce a self-supervision signal. In NOPD, the model learns from corrupted inputs while using its own predictions under clean inputs as token-level supervision. We show the effectiveness of NOPD on five visual reasoning tasks; it can match and even outperform reinforcement learning approaches or distillation from external models. Notably, when trained with 2.1K samples from Geometry3K, NOPD improves Qwen2.5-VL-7B by 20 points on its validation set. It also shows generalization on out-of-distribution test sets and achieves 7.4 point gains on MathVista. Furthermore, we demonstrate that NOPD is a general approach to enhance VLMs, achieving improvements across three models on 12 benchmarks.

1 INTRODUCTION

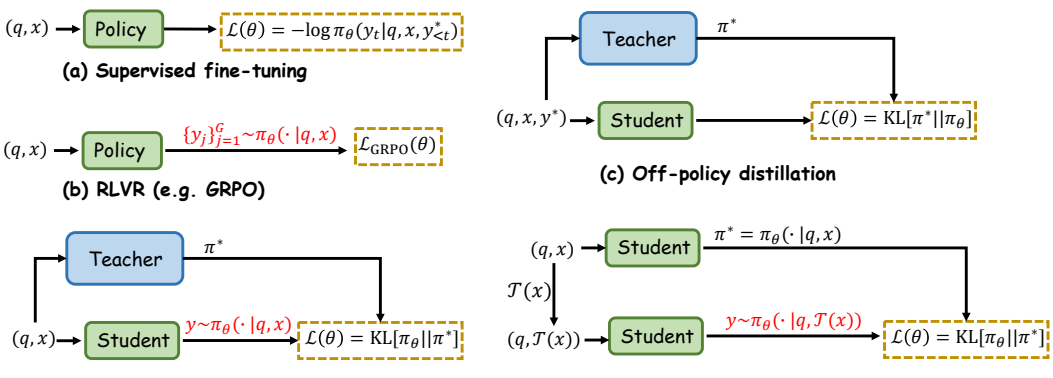
Vision-language models (VLMs) have made remarkable progress in visual question answering (VQA), visual reasoning and chart understanding (Bai et al., 2025b; Hong et al., 2025; Liu et al., 2023). The advancement is primarily achieved in the post-training stage, which may involve supervised fine-tuning (SFT) (Liu et al., 2023), reinforcement learning with human feedback (Ouyang et al., 2022) or verifiable rewards (RLVR) (Guo et al., 2025; OLMo et al., 2024; Shao et al., 2024), and knowledge distillation (Agarwal et al., 2024; Gu et al., 2024).

Although recent research has advanced the foundational capabilities of VLMs, typical post-training approaches exhibit inherent limitations. SFT usually needs human annotation or distillation from frontier models. RLVR typically requires verifiable questions and only provides a sparse reward signal for the whole response. Off-policy and on-policy knowledge distillation provide stronger and denser token-level supervision from an external model (usually called *teacher*).

Given these limitations and the strong capabilities of modern VLMs, we ask: can a VLM improve itself without any ground-truth answers or supervision from external models, i.e., "self-boosting"? Recently, some self-distillation approaches have been proposed to improve models without external models by using environment feedback (Hübotter et al., 2026), or using answers as additional context (Zhao et al., 2026). They are mainly designed for language models and still rely on external information. We aim to improve VLMs without any external supervision or feedback.

We show that supervision can be self-induced by exploiting prediction discrepancies under asymmetric input distributions. Specifically, when a model processes clean and corrupted inputs, the resulting difference in output distributions naturally provides a dense learning signal—without requiring labels, rewards, or external teachers. We instantiate this principle and propose noisy student on-policy self-distillation (NOPD), a self-improvement approach for VLMs. In NOPD, a single model plays both teacher and student roles. The teacher works with clean inputs, while the student is trained using corrupted ones. The predictions under clean inputs serve as token-level supervision for the noisy

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.



	No GT needed	On-policy data	No external models	Token-level learning signal	Low sampling cost
SFT	✘	✘	✔	✔	✔
RLVR	✘	✔	✔	✘	✘
KD	✔	✘	✘	✔	✔
OPD	✔	✔	✘	✔	✔
NOPD (ours)	✔	✔	✔	✔	✔

Figure 1: **Comparison of different post-training approaches for VLMs. Top**: demonstration of different approaches. q denotes text query and x denotes image. We omit the subscripts of loss functions except GRPO for simplicity. We highlight all sampling operations with red, e.g., $y \sim \pi_\theta(\cdot|q,x)$. π_θ denotes the policy model in (a) and (b), and the student model for distillation in (c), (d), and (e). π^* denotes the teacher for distillation methods. In (e), $\mathcal{T}$ denotes a pre-defined transformation. (e) gives an overview of our method NOPD. Different from typical on-policy distillation, NOPD shares the same parameters between student and teacher. The teacher supervision signal comes from the model under raw inputs (q,x). For the student model, inputs are corrupted images $\mathcal{T}(x)$ and text query q. **Bottom**: summary of comparison. GT denotes answers for given input questions, and may include detailed chain-of-thought (Wei et al., 2022b) traces.

inputs. Compared with previous methods, NOPD obviates the need for high-quality responses and external strong models as shown in Figure 1.

We conduct extensive experiments to validate the effectiveness of NOPD. We first conduct preliminary experiments to compare NOPD with different post-training methods on five visual reasoning benchmarks. Extensive experiments demonstrate that NOPD consistently improves VLMs across multiple benchmarks, achieving performance comparable to or exceeding methods that rely on external supervision. Notably, NOPD also leads to implicit reasoning compression, reducing output length while improving accuracy. These results suggest that effective post-training can be achieved without any external supervision. Finally, we validate whether NOPD generalizes across different models and tasks beyond reasoning.

2 Preliminaries

In this section, we briefly review existing post-training paradigms.

Notation. We use q to denote a text query, and x denotes an image from the training set $\mathcal{S}$. A vision-language model parameterized by θ is defined as π_θ. Given an input pair (q,x), its likelihood under π_θ is $\pi_\theta(y \mid q,x) = \prod_{t=1}^{|y|} \pi_\theta\left(y_t \mid q,x,y_{<t}\right)$, where $|y|$ denotes the number of tokens in y. y^* denotes the ground-truth response for input pair (q,x), which is typically obtained from human labeling or distillation from frontier models.

Supervised fine-tuning (SFT). SFT is widely used in post-training of modern foundation models that minimizes the following negative log-likelihood loss

$$\mathcal{L}_{\text{SFT}}(\theta) = -\sum_{t=1}^{|y^*|} \log \pi_\theta\left(y_t \mid q,x,y^*_{<t}\right). \tag{1}$$

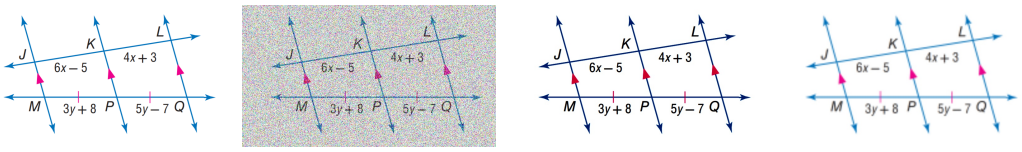


Figure 2: Examples of different distorted images. The types of image distortions (from left to right): original images, add noise, gamma transformation, downsample. We resize the downsampled image for clarity. The image is from the training set of Geometry3K.

Note that if y^* is generated by other models rather than human annotation, it is also called sequence-level knowledge distillation (Kim & Rush, 2016). SFT provides token-level supervision to guide the learning with the help of y^*.

Reinforcement learning with verifiable rewards (RLVR). RLVR (OLMo et al., 2024) is an on-policy reinforcement learning approach used for specific domains where questions have verifiable answers, such as math and code. The representative algorithm is GRPO (Shao et al., 2024), which computes advantages in a relative group instead of using additional critic models. The GRPO objective is formulated as

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{(q,x)\sim\mathcal{S},\,\{y_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q,x)}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\min\Big(w_{i,t}(\theta)\,\widehat{A}_{i,t},\right.$$
$$\left.\text{clip}\big(w_{i,t}(\theta),1-\varepsilon,1+\varepsilon\big)\widehat{A}_{i,t}\Big)\right], \tag{2}$$

where G is the number of generated responses for each input pair (q,x) (i.e., the group size) and ε is a hyperparameter to control the clipping range for stabilizing training. The importance ratio $w_{i,t}(\theta)$ and advantage $\widehat{A}_{i,t}$ of token $y_{i,t}$ are:

$$w_{i,t}(\theta) = \frac{\pi_\theta(y_{i,t}\mid q,x,y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t}\mid q,x,y_{i,<t})}, \quad \widehat{A}_{i,t} = \widehat{A}_i = \frac{r(q,x,y_i) - \text{mean}\left(\{r(q,x,y_i)\}_{i=1}^{G}\right)}{\text{std}\left(\{r(q,x,y_i)\}_{i=1}^{G}\right)}, \tag{3}$$

respectively, where all the tokens in y_i share the same advantage as $\widehat{A}_i$. $r(q,x,y_i)$ denotes the reward for given inputs (q,x) and responses y generated by π_θ. RLVR benefits from on-policy learning while suffering from sparse supervision signals and high sampling cost to rollout during training.

Off-policy knowledge distillation. The goal of knowledge distillation (KD) (Hinton et al., 2015) is to transfer knowledge from the strong model (usually named *teacher*) to the weak model (usually named *student*). Standard KD can be formulated as:

$$\mathcal{L}_{\text{KD}}(\theta) = \mathbb{E}_{(q,x)\sim\mathcal{S},\,y^*}\left[\mathcal{D}_{\text{KL}}\left(\pi^*(y^*\mid q,x)\,\|\pi_\theta(y^*\mid q,x)\right)\right], \tag{4}$$

where π^* denotes the teacher model and $\mathcal{D}_{\text{KL}}(P\|Q)$ denotes the Kullback–Leibler (KL) divergence.

On-policy distillation (OPD). OPD (Agarwal et al., 2024; Gu et al., 2024) trains the student model with self-generated sequences and learns feedback from the teacher on these sequences. Specifically, OPD minimizes an objective of the form:

$$\mathcal{L}_{\text{OPD}}(\theta) = \mathbb{E}_{(q,x)\sim\mathcal{S},\,y\sim\pi_\theta(\cdot|q,x)}\left[\mathcal{D}_{\text{KL}}\left(\pi_\theta(y\mid q,x)\,\|\pi^*(y\mid q,x)\right)\right]. \tag{5}$$

Off-/on-policy distillation provides dense token-level supervision without requiring human annotations, but typically relies on a stronger teacher model. When the teacher is derived from the same model as the student (e.g., sharing parameters) rather than an external model, we refer to this setting as on-policy self-distillation (Zhao et al., 2026; Hübotter et al., 2026). In this case, privileged information can be transferred through differences in inputs (Vapnik & Izmailov, 2015).

3 METHOD

We propose NOPD, short for **n**oisy student **o**n-**p**olicy self-**d**istillation, a self-training approach that enables VLMs to improve themselves without any ground-truth answers or external models.

Algorithm 1 Training Procedure of NOPD

Require: Training set $\mathcal{S} = \{(q_i, x_i)\}_{i=1}^{|S|}$; vision-language model π_θ; batch size B; image transformation $\mathcal{T}$; max iterations $T_{\max}$
Ensure: Trained model π_θ

```
Initialize t ← 0
while t < T_max and not converged do
    Sample a mini-batch {(q_i, x_i)}_{i=1}^B from S
    Sample on-policy response y_i ~ π_θ(· | q_i, T(x_i)),   i = 1, ..., B
    Compute the loss as Eq. (6)
        L(θ) = E_{(q,x)~S, y~π_θ(·|q,T(x))} [D_KL(π_θ(y | q, T(x)) ‖ π_θ(y | q, x))]
    Update model parameters θ using ∇_θ L (c.f. Eq. (8))
    t ← t + 1
end while
return π_θ
```

Algorithm 1 provides an overview of NOPD. The inputs to the student in the algorithm are corrupted images $\mathcal{T}(x)$ and the corresponding text query q. We first generate an on-policy response y under corrupted images $y \sim \pi_\theta(\mathcal{T}(x), q)$, where $\mathcal{T}$ is a pre-defined transformation for images. After that, we use the outputs of the same model π_θ under clean inputs (q, x) as the "teacher" output to provide dense supervision signals. Finally, our objective is to minimize the divergence between the student model and the teacher model as follows

$$\mathcal{L}_{\text{NOPD}}(\theta) = \mathbb{E}_{(q,x)\sim\mathcal{S},\, y\sim\pi_\theta(\cdot|q,\mathcal{T}(x))}\left[\mathcal{D}_{\text{KL}}\left(\pi_\theta(y \mid q, \mathcal{T}(x)) \,\|\, \underbrace{\pi_\theta(y \mid q, x)}_{\texttt{sg}}\right)\right], \tag{6}$$

where sg denotes stop gradient. Note that the KL divergence is not symmetric, which means that $\mathcal{D}_{\text{KL}}(P\|Q) \neq \mathcal{D}_{\text{KL}}(Q\|P)$, where $P = \pi_\theta(y \mid q, \mathcal{T}(x))$ and $Q = \pi_\theta(y \mid q, x)$. The KL divergence in Eq. (6) is called reverse KL. We consider two other commonly used variants: forward KL $\mathcal{D}_{\text{KL}}(Q\|P)$ and generalized Jensen–Shannon divergence (JSD), which is defined as

$$\mathcal{D}_{\text{JSD}}(P\|Q)(\beta) = \beta D_{\text{KL}}(P \,\|\, \beta P + (1-\beta)Q) + (1-\beta) D_{\text{KL}}(Q \,|\, \beta P + (1-\beta)Q), \tag{7}$$

where $\beta \in (0, 1)$ is the weighting coefficient. The gradients of JSD are similar to forward KL when $\beta \to 0$ and reverse KL when $\beta \to 1$, respectively. We verify the effectiveness of different divergences in Sec. 4.2. We stop the gradient from the "teacher" signal, i.e., $\pi_\theta(y \mid q, x)$, to prevent model collapse[1] and do not backpropagate through the student's sampling distribution $\pi_\theta(\cdot \mid q, \mathcal{T}(x))$, which makes the training process more stable and efficient following (Agarwal et al., 2024).

Distinguished from other post-training approaches, our method NOPD is a self-training approach to improve VLMs without any external supervision (models, verifiable answers, and reasoning traces). In addition, NOPD also has low sample cost, is more data efficient, and could reduce output length with performance increase; see Sec. 4.2 for verification. We compare our method with other post-training approaches in Figure 1 and provide more discussion in Sec. 5.

Our key contribution lies in introducing asymmetric input distributions by adding permutation $\mathcal{T}$ for the student vision inputs to enable self-induced supervision, which makes our method different from other self-distillation methods (Zhao et al., 2026; Hübotter et al., 2026). The motivation behind this is that we aim to improve student models by exposing them to difficult environments in terms of noise during learning. Learning from asymmetric input distributions to enable self-induced supervision also improves generalization to out-of-distribution domains.

[1] If we do not stop the gradient of $\pi_\theta(y \mid q, x)$, π_θ will generate the same output whatever the input to make the loss zero.

Table 1: **Comparison of different approaches.** We highlight the **best** result and performance degradation compared with the base model.

	Geometry3K		MathVista	MathVision	MMMU	MMMU-Pro
	test	val	testmini	test	val	overall
Qwen3.5-2B	73.2	65.3	76.1	44.3	64.2	50.3
SFT	73.2	66.0	74.8	42.8	63.8	49.4
KD	74.0	65.7	75.4	43.3	64.9	50.8
RLVR	**78.4**	68.2	79.3	**48.1**	67.0	53.5
OPD	77.2	67.8	78.7	46.6	**68.1**	**55.3**
NOPD (ours)	78.0	**69.1**	**80.4**	47.2	67.6	54.7
Qwen2.5-VL-7B	35.4	33.7	68.4	25.1	58.6	38.3
SFT	38.8	37.0	70.2	26.2	59.1	38.3
KD	39.1	37.3	70.2	25.8	59.1	38.1
RLVR	52.1	51.0	72.3	28.1	62.8	41.8
OPD	54.8	52.0	73.1	28.1	**63.6**	42.0
NOPD (ours)	**55.2**	**53.7**	**75.8**	**30.9**	63.2	**43.3**

Gradient Analysis. We can derive the NOPD gradient as follows (see Sec. B for details):

$$\nabla_\theta \mathcal{L}_{\text{NOPD}}(\theta) = \mathbb{E}_{(q,x)\sim\mathcal{S},\, y\sim\pi_\theta(\cdot|q,\mathcal{T}(x))}\left[\sum_{t=1}^{|y|}\mathbb{E}_{\hat{y}_t\sim\pi_\theta(\cdot|\mathcal{T}(x),q,y_{<t})}\right.$$
$$\left.\left[\log\frac{\pi_\theta(\hat{y}_t \mid q,\mathcal{T}(x))}{\pi_\theta(\hat{y}_t \mid q,x)}\cdot\nabla_\theta\log\pi_\theta(\hat{y}_t \mid q,\mathcal{T}(x),y_{<t})\right]\right], \tag{8}$$

where $\hat{y}_t \in \Delta^{|V|-1}$ is $|V|$-simplex and denotes the model's predicted probability distribution over the vocabulary at decoding step t, and $|V|$ is the vocabulary size.

The NOPD gradient can be interpreted as a contrastive policy gradient. The inner term, $\nabla_\theta \log \pi_\theta(\hat{y}_t \mid q, \mathcal{T}(x), y_{<t})$, is the standard policy gradient that propagates the sampled token $\hat{y}_t$ back to the model parameters. The log-ratio term $\log \frac{\pi_\theta(\hat{y}_t \mid q,\mathcal{T}(x))}{\pi_\theta(\hat{y}_t \mid q,x)}$ acts as a contrastive reward, amplifying updates for tokens whose predicted distributions differ significantly between the transformed input $(q, \mathcal{T}(x))$ and the original input (q, x).

The choice of image permutation. For image transformation $\mathcal{T}$, we consider three ways:

• Additive Gaussian noise. An augmented image x' is generated by adding Gaussian noise to the input image: $x' = \text{clip}(x + \epsilon, 0, 1)$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$. The noise standard deviation σ is randomly sampled from $\{0.08, 0.12, 0.18, 0.26, 0.38\}$, following (Hendrycks & Dietterich, 2019).

• Lower resolution. We randomly downsample the input image x by a factor of 2 or 4, resulting in a lower-resolution image. This process reduces image fidelity and decreases the number of vision tokens, thereby improving training efficiency.

• Gamma transformation. We apply gamma transformation to adjust the brightness and contrast of given images. To be specific, the augmented image is $x' = x^\gamma$ and we sample $\gamma \sim U[0.5, 2]$.

We provide examples of distorted images in Figure 2. Note that we choose hyperparameters for image transformation $\mathcal{T}$ directly widely used in computer vision tasks and do not tune hyperparameters due to the limitation of compute resources. During training, we randomly select one transformation and apply it to get corrupted image x.

4 Experiments

We conduct extensive experiments to answer the following questions:

• How does NOPD compare to other post-training approaches? (Sec. 4.1)

• How do the design choices affect performance? (Sec. 4.2)

Table 2: **Ablation studies** of key design choices. We train Qwen3.5-2B with the training set of Geometry3K and report accuracy on MathVista. Default settings are marked.

Loss	Acc.
Forward KL	78.1
Reverse KL	80.4
JSD ($\beta = 0.5$)	79.8

(a) Loss type

	Acc.
Full vocab	80.4
Sampled token	79.1

(b) Computation strategy

	Acc.
Fix	76.9
EMA	79.1
Sync	80.4

(c) Teacher update

Gen length	Acc.
1024	79.4
2048	80.4
4096	80.5

(d) Generation Length

• Is NOPD a general approach to improve VLMs itself in other domains? (Sec. 4.3)

4.1 A Preliminary Experiment on Reasoning Tasks

In this section, we conduct a preliminary experiment to verify the effectiveness of our method and compare it with other post-training approaches including SFT, off-policy KD, RLVR and OPD.

Models. We use Qwen2.5-VL-7B(-Instruct) (Bai et al., 2025b) and Qwen3.5-2B[2] as the base models. Note that Qwen3.5-2B is a hybrid model that supports thinking and no-thinking modes. To control for variables, we conduct experiments exclusively in the thinking mode.

Training details & baselines. We use the Geometry3K training set (Lu et al., 2021), which contains 2.1K image-text pairs. For the SFT baseline, we use responses distilled from Qwen3.5-27B/Qwen2.5-VL-72B-Instruct for Qwen3.5-2B/Qwen2.5-VL-7B. For a fair comparison, we keep the same teacher model for off-policy KD and on-policy distillation experiments. For RLVR, we use GRPO (Shao et al., 2024) as the standard implementation. We also experiment with other variants, such as DAPO (Yu et al., 2025) and GSPO (Zheng et al., 2025), but observe no significant improvement. Training hyperparameters are included in Table 7.

Evaluation. We first evaluate on the test and val split of Geometry3K (Lu et al., 2021) as the in-distribution test set. In addition, we add MathVista (Lu et al., 2024), MathVision (Wang et al., 2024a), MMMU (Yue et al., 2024) and MMMU-Pro (Yue et al., 2025) as out-of-distribution test sets. For Qwen2.5-VL-7B (Bai et al., 2025b), we use greedy decoding with CoT (Wei et al., 2022b) and report pass@1 accuracy. For Qwen3.5-2B, we follow the Qwen official guide to use random sampling and report avg@16 for stable performance. More details are included in Sec. A.1.

Results. Table 1 reports results on five benchmarks. First, NOPD consistently improves the base model across **all** benchmarks. Specifically, NOPD improves 4.4 points on MMMU-Pro and 4.8 points on the test set of Geometry3K with Qwen3.5-2B. In addition, NOPD achieves a 20-point improvement on the val split of Geometry3K and 7.4 points on MathVista with Qwen2.5-VL-7B. Trained only with the Geometry3K training set, NOPD also improves performance on out-of-distribution test sets, which demonstrates strong out-of-domain generalization.

Furthermore, NOPD can match or even outperform RLVR and OPD on multiple benchmarks. We also find that performance degradation often occurs with SFT and off-policy KD, which indicates that they struggle to improve open-weight models even with the help of teacher models or external labels. RLVR and OPD consistently improve all benchmarks for both models, but they require external ground-truth answers and models. The results show that NOPD is a self-training method that improves VLMs without any external signals (ground-truth answers or external models) and the effectiveness of NOPD matches RLVR and OPD.

4.2 Analysis

Design choices. We conduct extensive experiments to study key design choices in our method and present the results in Table 2. First, a key design choice is the loss function. We compare forward KL, reverse KL, and JSD in Table 2a. Reverse KL achieves the best result, while other objectives improve the baselines consistently.

[2] https://huggingface.co/Qwen/Qwen3.5-2B

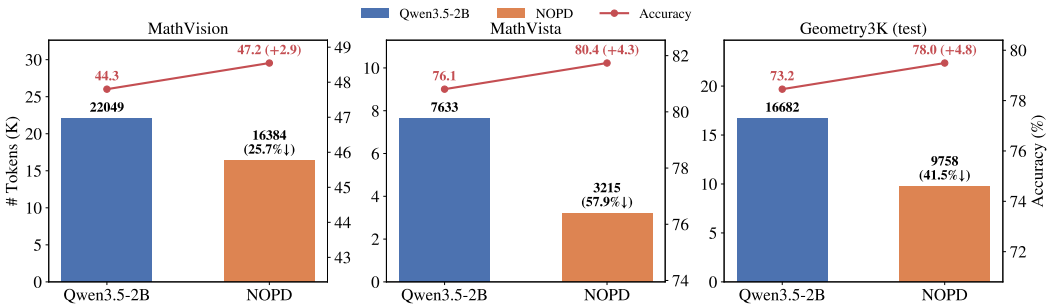


Figure 3: The number of output tokens and accuracy on three benchmarks.

When computing the divergence between the student and the teacher model, we compute all tokens over the vocabulary. Another strategy is to compute probabilities only at the tokens sampled by the student. Table 2b shows that computing probabilities on all tokens performs better, which demonstrates the benefit of richer supervision and teacher knowledge. Other reasons also include that only transferring knowledge of sampled tokens inevitably undermines the mathematical equivalence when we compute KL divergence, and preserving full vocabulary logits yields stable gradient estimates. However, directly transferring logits for a large vocabulary size (usually larger than 100K) becomes expensive. We only transfer last-layer teacher hidden states and recompute logits on the fly with the teacher's head when computing the loss function. Because the dimension of hidden states (e.g., 6144 for Qwen3.5-2B) is much smaller than the dimension of the full vocabulary (e.g., 248320 for Qwen3.5-2B), the corresponding communication overhead becomes affordable and practical, and only introduces negligible recomputation (Zhang et al., 2026).

In our experiments, we update the teacher fully in sync with the student model, i.e., $\pi^* = \pi_\theta$ at every training step. We also consider two other variants: (a) fixing the teacher model as the initial weights $\pi^* = \pi_\theta|_{t=0}$ where t denotes training steps, and (b) EMA update: $\pi^* \leftarrow m\pi^* + (1-m)\,\pi_\theta$ where $m = 0.9$ controls the update rate. We find that fully synchronizing the weights between the teacher and the student achieves the best performance in Table 2c. We hypothesize that the capability of the student model improves during training and can provide better supervision signals.

To reduce computational overhead, we constrain the generation length to 2048 for the student. We explore different generation lengths in Table 2d. We find that increasing the generation length to 4096 brings only limited gains compared with other baselines. To balance computation and performance, we set the maximum generation length to 2048. The intuition behind this is that early tokens are more critical for learning in the autoregressive decoding process. Errors in early tokens can accumulate and lead to incorrect outputs. Therefore, providing supervision signals only for early tokens is sufficient for the student's learning.

Implicit reasoning compression. We compare the number of output tokens and accuracy on three datasets using Qwen3.5-2B in Figure 3. Besides the improvement in accuracy, we find that our method simultaneously compresses the thinking process across three benchmarks, which improves the efficiency of reasoning. To be specific, on MathVista, our method reduces tokens by **57.9%** and improves accuracy by 4.3%. Although we do not design a specific technique for compressing tokens, NOPD achieves implicit reasoning compression, improving efficiency at test time.

Effectiveness across training datasets. To evaluate the effectiveness of our method across different training datasets, we conduct experiments on MMK12 (Meng et al., 2025), which contains 15.6K samples. All other settings remain unchanged, except that we increase the training steps to 200 due to the larger dataset size compared to Geometry3K. As shown in Figure 4, NOPD consistently improves performance when trained on MMK12, demonstrating strong scalability with increased data and training compute. Specially, accuracy on MathVision improves by 3.2 points, from 47.2% to 50.4%.

Corruption type of images. We conduct experiments with different image corruption types to verify their importance. As shown in Table 3, all corruptions improve the base model, with adding noise yielding the largest gain (73.2 $\rightarrow$ 77.1) and combining all types performing best. This highlights the importance of image corruption for student inputs.

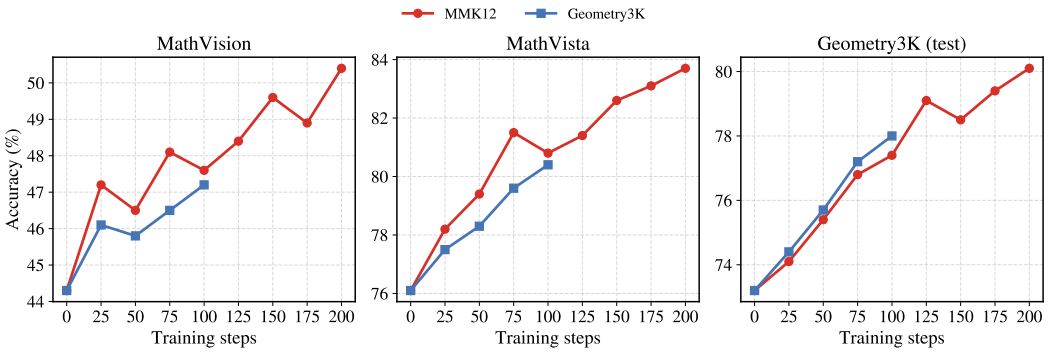


Figure 4: **Accuracy on three benchmarks during training with two different training sets**. We train Qwen3.5-2B with MMK12 and Geometry3K, respectively.

4.3 Could NOPD be a General Approach to Improve VLMs?

Models. To validate whether our method can be used to improve VLMs on different tasks and to assess its scalability with data, we conduct experiments using larger-scale datasets and evaluate on multiple tasks. We choose Qwen2.5-VL-7B (Bai et al., 2025b) as the instruction model and Qwen3-VL-Thinking (Bai et al., 2025a) as the thinking model. To further examine the effectiveness of our method across different models, we include GLM4.6-V-Flash with 9B parameters (Hong et al., 2025).

Table 3: Ablation on corruption. N, G, L denotes noise, gamma transformation, lower resolution, respectively. We report results on Geometry3K test set with Qwen3.5-2B.

N	G	L	Geometry3K
✔	✔	✔	78.0
✔			77.1
	✔		74.4
		✔	76.8
			73.6
w/o training			73.2

Training & data. We collect data from multiple sources, including Geometry3K (Lu et al., 2021), GeomVerse (Kazemi et al., 2023), Geo170K (Gao et al., 2023), MMK12 (Meng et al., 2025), GeoQA+ (Cao & Xiao, 2022), CoSyn (Yang et al., 2025b), VQAv2 (Goyal et al., 2017), ArxivQA (Li et al., 2024), ChartQA (Masry et al., 2022) and GQA (Hudson & Manning, 2019).

We remove similar images or questions to improve the diversity of the training set. In addition, to ensure vision relevance so that all questions must be answered with images, we prompt Qwen3-VL-8B-Instruct and remove image-irrelevant questions. To balance the training data distribution, we classify all questions into three domains: general, reasoning, and chart, and uniformly sample different domains during training. We get 19K/56K/54K samples for general, chart, and reasoning domains, respectively. We increase training steps by 1000 and keep other hyperparameters the same.

Evaluation. To provide a comprehensive evaluation across diverse tasks, we assess all models on 12 benchmarks covering three categories, including general visual question answering (VQA), reasoning, and chart understanding. Specifically, for general VQA, we adopt MMStar (Chen et al., 2024), MMBench-EN (Liu et al., 2024), BLINK (Fu et al., 2024), CV-Bench (Tong et al., 2024), and Realworld-QA.[3] For reasoning tasks, we utilize MathVista (Lu et al., 2024), MathVision (Wang et al., 2024a), MMMU (Yue et al., 2024), and MMMU-Pro (Yue et al., 2025). For chart understanding, we evaluate models on ChartQA-Pro (Masry et al., 2025), ChartMuseum (Tang et al., 2025), and CharXiv (reasoning) (Wang et al., 2024b).

For different models, we adopt decoding parameters recommended by the official guide (see Table 6) and follow the evaluation protocol in `lmms-eval` (Zhang et al., 2025).

Results. We present results in Table 4. First, it is noticed that NOPD achieves consistent gains across three models. Specifically, we improve Qwen2.5-VL-7B on **all** benchmarks. For Qwen3-VL-8B-Thinking and GLM-4.6V-Flash, we get improvements on 10 out of 12 benchmarks. This shows that NOPD could be used as a general self-improvement approach for VLMs. Furthermore, NOPD

[3] https://huggingface.co/datasets/xai-org/RealworldQA

Table 4: **Empirical results on multimodal benchmarks with different models.** +x and -x indicate improvement/decline compared with initial models.

		Qwen2.5 -VL-7B	NOPD -7B	Qwen3-VL -8B-Thinking	NOPD -8B	GLM-4.6V -Flash	NOPD -9B
General VQA	MMStar	62.5	69.9 (+7.4)	73.6	75.2 (+1.6)	74.3	76.8 (+2.5)
	MMBench-EN	83.4	83.6 (+0.2)	90.6	90.4 (-0.2)	91.0	90.8 (-0.2)
	BLINK	56.4	62.3 (+5.9)	64.7	64.1 (-0.6)	65.5	66.4 (+0.9)
	CV-Bench	80.0	84.3 (+4.3)	84.8	85.6 (+0.8)	86.0	86.3 (+0.3)
	Realworld-QA	68.5	73.5 (+5.0)	71.9	73.4 (+1.5)	76.9	76.9 (+0.0)
Reasoning	MathVista	68.6	78.4 (+9.8)	78.5	81.4 (+2.9)	82.9	83.3 (+0.4)
	MathVision	22.4	34.4 (+12.0)	59.6	62.1 (+2.5)	54.1	57.5 (+3.4)
	MMMU	51.3	58.1 (+6.8)	73.5	75.7 (+2.2)	71.2	74.2 (+3.0)
	MMMU-Pro	34.6	39.6 (+5.0)	60.9	62.6 (+1.7)	59.9	63.7 (+3.8)
Chart	ChartQA-Pro	43.3	44.3 (+1.0)	58.4	62.9 (+4.5)	62.6	62.4 (-0.2)
	ChartMuseum	26.8	31.4 (+4.6)	46.7	51.6 (+4.9)	49.8	51.3 (+1.5)
	CharXiv	42.5	47.5 (+5.0)	53.0	58.4 (+5.4)	59.6	63.5 (+3.9)

yields substantial improvements across all three domains, which shows the effectiveness of NOPD is not limited to reasoning tasks but extends to other domains such as VQA and chart understanding.

5 Related Work

Self-improvement for foundation models. Previous work has advanced the self-improvement of foundation models by fine-tuning on self-generated data (Zelikman et al., 2022), automatically constructing preference data (Deng et al., 2024; Dong et al., 2025), or using LLMs as judges to select responses or label preference data (Yuan et al., 2024; Zhang et al., 2024; Gulcehre et al., 2023). Recently, some self-distillation approaches (He et al., 2026; Zhao et al., 2026), where a single model plays both teacher and student roles in the on-policy distillation process, have been proposed to improve itself without additional teachers. However, the above methods are designed for language models and require additional feedback from environments (Hübotter et al., 2026) or ground-truth labels (Zhao et al., 2026). Our NOPD also belongs to self-distillation approaches but does not need additional feedback or answers to improve VLMs.

Vision-language models. VLMs that combine visual encoders such as CLIP (Radford et al., 2021) with large language models have shown strong ability to understand and reason with both visual and text information inputs (Liu et al., 2023; Bai et al., 2025b). Recently, the development of VLMs has largely been driven by a relentless pursuit of scale in terms of model and data size (Doshi, 2025; Hong et al., 2025; Bai et al., 2025a; Team et al., 2026). Post-training with high-quality SFT data and large-scale reinforcement learning plays an important role. However, this leads to a high demand for human labels, distillation from frontier models, and the high sampling cost of reinforcement learning. To tackle this, we present NOPD, a self-training approach that eliminates the need for external supervision signals.

Post-training approaches. Post-training that fine-tunes the pre-trained language models aims to make pre-trained models follow human instructions and give useful and helpful responses (Ouyang et al., 2022). Typical post-training includes supervised fine-tuning (Wei et al., 2022a), direct preference optimization (Rafailov et al., 2023), reinforcement learning with verifiable rewards (Shao et al., 2024; Liu et al., 2025) or from human feedback (Ouyang et al., 2022), and distillation from stronger models (Agarwal et al., 2024; Gu et al., 2024; Kim & Rush, 2016).

6 Conclusion

We introduce Noisy Student On-Policy Self-Distillation (NOPD), a simple yet effective approach to improve VLMs without requiring ground-truth answers or external models. Our key insight is that forcing models to learn in difficult environments, e.g., noisy images, improves their capability across different tasks. Unlike traditional distillation approaches that require teacher models, we use the outputs of the same model under clean inputs as the dense token-level supervision signal. Experimental results show that NOPD improves the reasoning ability of VLMs across both in-distribution and out-of-distribution sets. Furthermore, we demonstrate that NOPD can serve as a general approach to improve VLMs across different models and tasks.

AI USE STATEMENT

In this study, generative AI tools were employed exclusively for language refinement and coding support. More specifically, GPT was utilized during the writing phase to enhance the manuscript's clarity and rectify grammatical mistakes. Codex served as a coding assistant throughout the experimental stage, helping with code generation. These AI tools were not involved in developing research concepts, establishing scientific assertions or conclusions, or analyzing experimental outcomes.

All content produced with AI assistance underwent thorough review and validation by the authors. In particular, every piece of AI-generated code was manually examined by the author team, and its accuracy was confirmed by running it and reviewing the outputs. The full manuscript, including any AI-assisted language modifications, was manually checked by the authors to guarantee technical precision and alignment with the intended meaning. All experimental findings and conclusions were independently confirmed by the author team.

We bear full responsibility for the final version of this work, encompassing all text, claims, code, experimental results, and other materials created with the help of generative AI.

REFERENCES

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

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Jie Cao and Jing Xiao. An augmented benchmark dataset for geometric question answering through dual parallel text encoding. In *Proceedings of the 29th international conference on computational linguistics*, pp. 1511–1520, 2022.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are we on the right way for evaluating large vision-language models? In *NeurIPS*, 2024.

Kanzhi Cheng, Wenpo Song, Jiaxin Fan, Zheng Ma, Qiushi Sun, Fangzhi Xu, Chenyang Yan, Nuo Chen, Jianbing Zhang, and Jiajun Chen. Caparena: Benchmarking and analyzing detailed image captioning in the llm era. In *ACL*, 2025.

Yihe Deng, Pan Lu, Fan Yin, Ziniu Hu, Sheng Shen, Quanquan Gu, James Zou, Kai-Wei Chang, and Wei Wang. Enhancing large vision language models with self-training on image comprehension. In *NeurIPS*, 2024.

Qingxiu Dong, Li Dong, Xingxing Zhang, Zhifang Sui, and Furu Wei. Self-boosting large language models with synthetic preference data. In *ICLR*, 2025.

Rohan Doshi. Gemini 3 pro: the frontier of vision ai. https://blog.google/innovation-and-ai/technology/developers-tools/gemini-3-pro-vision/, December 2025. Google Blog.

Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A Smith, Wei-Chiu Ma, and Ranjay Krishna. Blink: Multimodal large language models can see but not perceive. In *ECCV*, 2024.

Jiahui Gao, Renjie Pi, Jipeng Zhang, Jiacheng Ye, Wanjun Zhong, Yufei Wang, Lanqing Hong, Jianhua Han, Hang Xu, Zhenguo Li, et al. G-llava: Solving geometric problem with multi-modal large language model. *arXiv preprint arXiv:2312.11370*, 2023.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *CVPR*, 2017.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *ICLR*, 2024.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, et al. Reinforced self-training (rest) for language modeling. *arXiv preprint arXiv:2308.08998*, 2023.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Yinghui He, Simran Kaur, Adithya Bhaskar, Yongjin Yang, Jiarui Liu, Narutatsu Ri, Liam Fowl, Abhishek Panigrahi, Danqi Chen, and Sanjeev Arora. Self-distillation zero: Self-revision turns binary rewards into dense supervision. *arXiv preprint arXiv:2604.12002*, 2026.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR*, 2019.

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

Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, et al. Glm-4.5 v and glm-4.1 v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning. *arXiv preprint arXiv:2507.01006*, 2025.

Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, et al. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*, 2026.

Drew A. Hudson and Christopher D. Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *CVPR*, 2019.

Mehran Kazemi, Hamidreza Alvari, Ankit Anand, Jialin Wu, Xi Chen, and Radu Soricut. Geomverse: A systematic evaluation of large models for geometric reasoning. *arXiv preprint arXiv:2312.12241*, 2023.

Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. In *EMNLP*, 2016.

Lei Li, Yuqi Wang, Runxin Xu, Peiyi Wang, Xiachong Feng, Lingpeng Kong, and Qi Liu. Multimodal arxiv: A dataset for improving scientific comprehension of large vision-language models. In *ACL*, 2024.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023.

Xiangyan Liu, Jinjie Ni, Zijian Wu, Chao Du, Longxu Dou, Haonan Wang, Tianyu Pang, and Michael Qizhe Shieh. Noisyrollout: Reinforcing visual reasoning with data augmentation. In *NeurIPS*, 2025.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. Mmbench: Is your multi-modal model an all-around player? In *ECCV*, 2024.

Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning. In *ACL*, 2021.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *ICLR*, 2024.

Ahmed Masry, Xuan Long Do, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. In *ACL*, 2022.

Ahmed Masry, Mohammed Saidul Islam, Mahir Ahmed, Aayush Bajaj, Firoz Kabir, Aaryaman Kartha, Md Tahmid Rahman Laskar, Mizanur Rahman, Shadikur Rahman, Mehrad Shahmohammadi, et al. Chartqapro: A more diverse and challenging benchmark for chart question answering. In *ACL*, 2025.

Fanqing Meng, Lingxiao Du, Zongkai Liu, Zhixiang Zhou, Quanfeng Lu, Daocheng Fu, Tiancheng Han, Botian Shi, Wenhai Wang, Junjun He, et al. Mm-eureka: Exploring the frontiers of multimodal reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2503.07365*, 2025.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*, 2024.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, 2023.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Liyan Tang, Grace Kim, Xinyu Zhao, Thom Lake, Wenxuan Ding, Fangcong Yin, Prasann Singhal, Manya Wadhwa, Zeyu Leo Liu, Zayne Sprague, et al. Chartmuseum: Testing visual reasoning capabilities of large vision-language models. *arXiv preprint arXiv:2505.13444*, 2025.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, SH Cai, Yuan Cao, Y Charles, HS Che, Cheng Chen, Guanduo Chen, et al. Kimi k2. 5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*, 2026.

Shengbang Tong, Ellis L Brown II, Penghao Wu, Sanghyun Woo, ADITHYA JAIRAM IYER, Sai Charitha Akula, Shusheng Yang, Jihan Yang, Manoj Middepogu, Ziteng Wang, Xichen Pan, Rob Fergus, Yann LeCun, and Saining Xie. Cambrian-1: A fully open, vision-centric exploration of multimodal LLMs. In *NeurPS*, 2024.

Vladimir Vapnik and Rauf Izmailov. Learning using privileged information: Similarity control and knowledge transfer. *Journal of Machine Learning Research*, 16(61):2023–2049, 2015.

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with MATH-vision dataset. In *NeurIPS*, 2024a.

Zirui Wang, Mengzhou Xia, Luxi He, Howard Chen, Yitao Liu, Richard Zhu, Kaiqu Liang, Xindi Wu, Haotian Liu, Sadhika Malladi, Alexis Chevalier, Sanjeev Arora, and Danqi Chen. Charxiv: Charting gaps in realistic chart understanding in multimodal LLMs. In *NeurIPS*, 2024b.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *ICLR*, 2022a.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022b.

Shijia Yang, Yunong Liu, Bohan Zhai, Ximeng Sun, Zicheng Liu, Emad Barsoum, Manling Li, and Chenfeng Xu. Captionqa: Is your caption as useful as the image itself? *arXiv preprint arXiv:2511.21025*, 2025a.

Yue Yang, Ajay Patel, Matt Deitke, Tanmay Gupta, Luca Weihs, Andrew Head, Mark Yatskar, Chris Callison-Burch, Ranjay Krishna, Aniruddha Kembhavi, and Christopher Clark. Scaling text-rich image understanding via code-guided synthetic multimodal data generation. In *ACL*, 2025b.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, YuYue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *NeurIPS*, 2025.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason E Weston. Self-rewarding language models. In *ICML*, 2024.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *CVPR*, 2024.

Xiang Yue, Tianyu Zheng, Yuansheng Ni, Yubo Wang, Kai Zhang, Shengbang Tong, Yuxuan Sun, Botao Yu, Ge Zhang, Huan Sun, Yu Su, Wenhu Chen, and Graham Neubig. Mmmu-pro: A more robust multi-discipline multimodal understanding benchmark. In *ACL*, 2025.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. STar: Bootstrapping reasoning with reasoning. In *NeurIPS*, 2022.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. ReST-MCTS*: LLM self-training via process reward guided tree search. In *NeurIPS*, 2024.

Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, et al. Lmms-eval: Reality check on the evaluation of large multimodal models. In *NAACL*, 2025.

Songming Zhang, Xue Zhang, Tong Zhang, Bojie Hu, Yufeng Chen, and Jinan Xu. Kdflow: A user-friendly and efficient knowledge distillation framework for large language models. *arXiv preprint arXiv:2603.01875*, 2026.

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.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

A Experiment Details

A.1 Evaluation Details

Decoding & judge. For all models, we use vLLM to accelerate inference and follow the official guide to set the decoding parameters and summarize them in Table 6. For Qwen2.5-VL-7B, we report pass@1 accuracy because we use greedy decoding. For other models, we report avg@16 to provide stable results. For tasks that need the language model to extract and judge answers, we use GPT-OSS-120B.[4]

Benchmark-specific choices. In the preliminary experiments of Sec. 4.1, we evaluate the test/validation split of Geometry3K in an open-ended manner, which differs from the multiple-choice evaluation in `lmms-eval`. For MMMU and MMMU-Pro with Qwen2.5-VL-7B, we report the maximum performance between the direct answer and the CoT prompt, following the official guide. For ChartMuseum, we report results on the test set. For CharXiv, we report results on the validation set for the reasoning split.

A.2 Training Details

We summarize the training hyperparameters in Table 7.

B More Results and Analysis

Comparison with OPSD. OPSD (Zhao et al., 2026) also belongs to on-policy self-distillation where a single model plays both teacher and student roles. OPSD uses the ground-truth solution as additional privileged information input to the teacher model to provide a high-quality supervision signal. Formally, we reformat it in training VLM scenarios:

$$\mathcal{L}_{\text{OPSD}}(\theta) = \mathbb{E}_{(q,x)\sim\mathcal{D},\, y\sim\pi_\theta(\cdot|q,x)}\left[\mathcal{D}_{\text{KL}}\left(\pi_\theta\left(y \mid q, x\right) \| \pi_\theta\left(y \mid q, x, y^*\right)\right)\right]. \tag{9}$$

Unlike OPSD, our method NOPD does not need ground-truth solutions to provide additional information. In addition, OPSD and NOPD are also orthogonal, which means we could combine OPSD and NOPD. We conduct preliminaries following the section in Sec. 4.1 and results are shown as follows.

Table 5: **Comparison with OPSD.**

	Geometry3K		**MathVista**	**MathVision**	**MMMU**	**MMMU-Pro**
	test	val	testmini	test	val	overall
Qwen3.5-2B	73.2	65.3	76.1	44.3	64.2	50.3
OPSD	76.4	67.8	79.5	44.9	66.1	50.5
NOPD	78.0	69.1	**80.4**	47.2	67.6	54.7
OPDS+NOPD	**78.5**	**70.4**	80.2	**47.5**	**68.8**	**55.6**

From results of Table 5, we can see that NOPD is more effective than OPSD while eliminating the need for ground-truth solutions. Furthermore, combining OPSD and NOPD could enhance performance.

Difference with NoisyRollout. NOPD and NoisyRollout (Liu et al., 2025) share a similar part of adding corruption to images for VLMs. However, the motivations behind this are totally different. NoisyRollout is a variant of GRPO that adds noise to images to *improve exploration in reinforcement learning*. NoisyRollout still belongs to RLVR and needs a verifiable answer as we compare in Figure 1. Our NOPD uses discrepancies between clean and distorted images as a self-supervision signal to improve VLMs without an external model or answers.

[4] https://huggingface.co/openai/gpt-oss-120b

Unleashing the potential of NOPD for open-ended generation tasks. We mainly evaluate NOPD on verifiable tasks, such as reasoning and question answering. Since on-policy distillation provides a *dense* token-level supervision signal, it is natural to ask: is NOPD effective on open-ended generation tasks, such as image captioning? We conduct preliminary experiments on image captioning tasks. We randomly sample 50K images from ShareGPT-4o[5] and train Qwen2.5-VL-7B for 200 steps.

We evaluate the ability of image captioning on CaptionQA (Yang et al., 2025a) and CapArena (Cheng et al., 2025). We report accuracy on CaptionQA and win rate against Gemini-3.0-Flash and GPT-5 on CapArena with 100 samples. Results on CaptionQA are shown in Table 8. From Table 8, it can be seen that NOPD improves Qwen2.5-VL-7B on all domains. Specifically, in natural domains, NOPD achieves 13-point gains. We also use GPT-4o as a VLM judge against GPT-5 and Gemini-3-Flash in Figure 5 on the CapArena Benchmarks. With NOPD, a model with 7B parameters can match the performance of frontier models. The quality of captions on more than 50% of images outperforms and matches proprietary models such as Gemini-3-Flash and GPT-5.

Gradient derivation. First, we detail the gradient of the NOPD loss as defined in Eq. (6):

$$\nabla_\theta \mathcal{L}_{\text{NOPD}}(\theta) = \nabla_\theta \sum_{t=1}^{|y|} \sum_{v=1}^{|V|} \pi_\theta(y_{v,t}|q, \mathcal{T}(x)) \log \frac{\pi_\theta(y_{v,t}|q, \mathcal{T}(x))}{\pi_\theta(y_{v,t}|q, x)}. \tag{10}$$

Since we stop the gradient of $\pi_\theta(y_{v,t}|q, x)$, we denote $\pi_{\text{ref}}(y_{v,t}|q, x) = \pi_\theta(y_{v,t}|q, x)$ for clarity and ignore the subscript NOPD.

$$\begin{aligned}
\nabla_\theta \mathcal{L}(\theta) &= \nabla_\theta \sum_{t=1}^{|y|} \sum_{v=1}^{|V|} \pi_\theta(y_{v,t}|q, \mathcal{T}(x)) \log \frac{\pi_\theta(y_{v,t}|q, \mathcal{T}(x))}{\pi_{\text{ref}}(y_{v,t}|q, x)} \\
&= \sum_{t=1}^{|y|} \sum_{v=1}^{|V|} \log \frac{\pi_\theta(y_{v,t}|q, \mathcal{T}(x))}{\pi_{\text{ref}}(y_{v,t}|q, x)} \nabla_\theta \pi_\theta(y_{v,t}|q, \mathcal{T}(x)) + \sum_{t=1}^{|y|} \sum_{v=1}^{|V|} \pi_\theta(y_{v,t}|q, \mathcal{T}(x)) \nabla_\theta \log \frac{\pi_\theta(y_{v,t}|q, \mathcal{T}(x))}{\pi_{\text{ref}}(y_{v,t}|q, x)} \\
&= \sum_{t=1}^{|y|} \sum_{v=1}^{|V|} \log \frac{\pi_\theta(y_{v,t}|q, \mathcal{T}(x))}{\pi_{\text{ref}}(y_{v,t}|q, x)} \nabla_\theta \pi_\theta(y_{v,t}|q, \mathcal{T}(x)) + \sum_{t=1}^{|y|} \sum_{v=1}^{|V|} \pi_\theta(y_{v,t}|q, \mathcal{T}(x)) \frac{\nabla_\theta \pi_\theta(y_{v,t}|q, \mathcal{T}(x))}{\pi_\theta(y_{v,t}|q, \mathcal{T}(x))} \\
&= \sum_{t=1}^{|y|} \sum_{v=1}^{|V|} \log \frac{\pi_\theta(y_{v,t}|q, \mathcal{T}(x))}{\pi_{\text{ref}}(y_{v,t}|q, x)} \nabla_\theta \pi_\theta(y_{v,t}|q, \mathcal{T}(x)) + \sum_{t=1}^{|y|} \sum_{v=1}^{|V|} \nabla_\theta \pi_\theta(y_{v,t}|q, \mathcal{T}(x)) \\
&= \sum_{t=1}^{|y|} \sum_{v=1}^{|V|} \log \frac{\pi_\theta(y_{v,t}|q, \mathcal{T}(x))}{\pi_{\text{ref}}(y_{v,t}|q, x)} \nabla_\theta \pi_\theta(y_{v,t}|q, \mathcal{T}(x)) + \sum_{t=1}^{|y|} \nabla_\theta \sum_{v=1}^{|V|} \underbrace{\pi_\theta(y_{v,t} \mid q, \mathcal{T}(x))}_{1} \\
&= \sum_{t=1}^{|y|} \sum_{v=1}^{|V|} \log \frac{\pi_\theta(y_{v,t}|q, \mathcal{T}(x))}{\pi_{\text{ref}}(y_{v,t}|q, x)} \nabla_\theta \pi_\theta(y_{v,t}|q, \mathcal{T}(x)) \\
&= \sum_{t=1}^{|y|} \sum_{v=1}^{|V|} \log \frac{\pi_\theta(y_{v,t}|q, \mathcal{T}(x))}{\pi_{\text{ref}}(y_{v,t}|q, x)} \pi_\theta(y_{v,t}|q, \mathcal{T}(x)) \nabla_\theta \log \pi_\theta(y_{v,t}|q, \mathcal{T}(x)) \\
&= \sum_{t=1}^{|y|} \mathbb{E}_{\hat{y}_t \sim \pi_\theta(\cdot|\mathcal{T}(x), q, y_{<t})} \left[\log \frac{\pi_\theta(\hat{y}_t \mid q, \mathcal{T}(x))}{\pi_\theta(\hat{y}_t \mid q, x)} \cdot \nabla_\theta \log \pi_\theta(\hat{y}_t \mid q, \mathcal{T}(x), y_{<t}) \right].
\end{aligned} \tag{11}$$

Finally, we get the gradient of $\mathcal{L}_{\text{NOPD}}$ in Eq. (8).

Does NOPD work by improving diversity? We found that injecting vision permutation $\mathcal{T}$ slightly changes the distribution of student models. This naturally raises the question: does the improvement of NOPD come from the diversity of rollouts? To disentangle the effects of diversity and vision permutation, we conduct experiments under different rollout temperatures. For each question, we

[5] https://sharegpt4o.github.io/

Table 6: Summary of decoding parameters during evaluation.

Model	Temperature	Top-p	Top-k	Presence penalty	Repetition penalty	Max tokens
Qwen2.5-VL-7B	0	1.0	1	1.0	1.0	2048
Qwen3.5-2B	1.0	0.95	20	1.5	1.0	32768
Qwen3-VL-8B-Thinking	1.0	0.95	20	0.0	1.0	40960
GLM-4.6V-Flash	0.8	0.6	2	1.0	1.1	16384

Table 7: Training hyperparameters. "X/Y" denotes that we use X for Qwen2.5-VL-7B and Y for Qwen3.5-2B.

	SFT	Off-polocy KD	RLVR	OPD	NOPD
Training backend			FSDP2		
Precision			BF16		
Gradient clip			1.0		
Attention backend			FlashAttention 2		
Optimizer			AdamW		
Learning rate			1e-6		
Inference engine	-	-		SGLang	
LRScheduler	cosine	cosine	constant	constant	constant
Epoch / iter	3 epoch	3 epoch	100 steps	100 steps	100 steps
Global batch size	64	64	128	32	32
Rollout temperature	-	-	1.0	1.0	1.0
Rollout top-p	-	-	0.95	0.95	0.95
Rollout number	-	-	8	1	1
Max gen length	-	-	2048/16384	2048	2048

sample 5 responses and score their diversity using an LLM as a judge on a scale from 1 to 10. Note that we keep the number of rollout response to 1 when computing gradients to align other settings. Other questions are only used to examine diversity.

Results are shown in Table 9. We found that with image transformation $\mathcal{T}$, the diversity of rollout outputs does not improve at temperatures of 0.8 and 1.0, yet performance increases. At a temperature of 1.2, diversity continues to improve but performance drops. These results indicate that the performance gains of NOPD do not stem from higher diversity.

Visualization examples. We provide two example visualizations in Table 10 and Table 11 to demonstrate that NOPD improves the perception and reasoning abilities compared with the base model. In particular, Qwen2.5-VL-7B fails to properly analyze the relative relationships between the line segments in the figure (Table 10) and to correctly identify which segment the labeled length refers to (Table 11).

Table 8: Results on CaptionQA.

Model	Natural	Document	E-comm	Embodied	Avg.
GPT5	88.15	89.98	95.44	85.30	89.72
Gemini-3-Flash	87.66	84.59	89.97	84.03	86.56
Gemini-3.1-Pro	88.49	90.82	90.36	85.15	88.71
Qwen2.5-VL-7B	71.64	75.85	85.38	68.36	75.31
+ NOPD	84.68	82.70	87.64	74.19	82.31

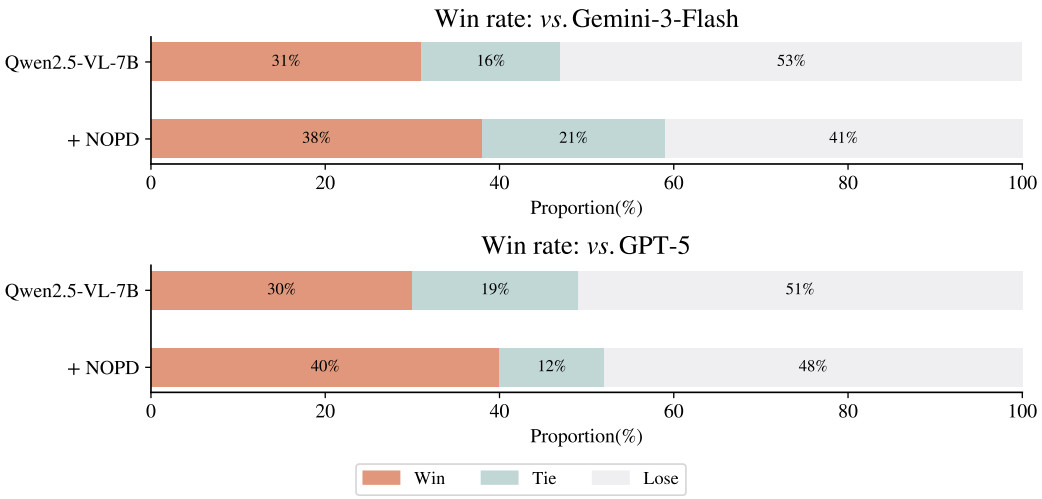
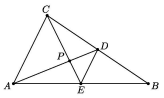


Figure 5: Win-rate comparison on CapArena with Gemini-3-Flash and GPT-5.

Table 9: Ablation on different rollout temperature.

	Temperature	Accuracy	Diversity
w/o $\mathcal{T}$	0.8	73.1	7.2
	1.0	73.6	7.4
	1.2	73.0	7.5
w/ $\mathcal{T}$	0.8	78.0	7.1
	1.0	78.0	7.4
	1.2	77.8	7.8

Table 10: Case study of a visualized example from MathVision with Qwen2.5-VL-7B. We highlight errors generated by Qwen2.5-VL-7B.



Question: In $\triangle ABC$, medians $\overline{AD}$ and $\overline{CE}$ intersect at P, $PE = 1.5$, $PD = 2$, and $DE = 2.5$. What is the area of $AEDC$? Answer: 13.5

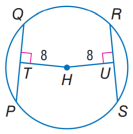
Qwen2.5-VL-7B

To solve the problem, we need to determine the area of quadrilateral $AEDC$ in $\triangle ABC$ given the lengths of the segments formed by the intersection of the medians at point P. 1. **Identify the properties of the medians and centroid:** - The medians of a triangle intersect at the centroid P, which divides each median into a ratio of 2:1. - Therefore, $AP : PD = 2 : 1$ and $CP : PE = 2 : 1$. 2. **Determine the lengths of the segments:** - Given $PE = 1.5$ and $PD = 2$, we can find PD and PE in terms of the total length of the median. - Since $PD = 2$ and $PE = 1.5$, the total length of the median CE is $PD + PE = 2 + 1.5 = 3.5$. - The centroid P divides CE into segments CP and PE in the ratio 2:1. Therefore, $CP = 2 \times 1.5 = 3$ and $PE = 1.5$. 3. **Calculate the area of $\triangle AEDC$:** - The area of $\triangle AEDC$ can be found by considering the areas of the smaller triangles formed by the medians. - The centroid divides the triangle into six smaller triangles of equal area. - The area of $\triangle AEDC$ is the sum of the areas of $\triangle APE$ and $\triangle CPE$. 4. **Use the given lengths to find the area:** - The area of $\triangle AEDC$ is the sum of the areas of $\triangle APE$ and $\triangle CPE$. - The area of $\triangle APE$ is $\frac{1}{3}$ of the area of $\triangle AEC$ and the area of $\triangle CPE$ is $\frac{1}{3}$ of the area of $\triangle CED$. - The area of $\triangle AEDC$ is the sum of the areas of $\triangle APE$ and $\triangle CPE$, which is $\frac{2}{3}$ of the area of $\triangle AEC$. 5. **Calculate the area of $\triangle AEDC$:** - The area of $\triangle AEDC$ is $\frac{2}{3}$ of the area of $\triangle AEC$. - The area of $\triangle AEC$ is the sum of the areas of $\triangle APE$ and $\triangle CPE$, which is $\frac{2}{3}$ of the area of $\triangle AEDC$. 6. **Final answer:** - The area of $\triangle AEDC$ is $\boxed{10}$. ✘

NOPD

To solve the problem, we aim to find the area of quadrilateral $AEDC$ in $\triangle ABC$ based on the segment lengths determined by the intersection of the medians at point P.
1. **Identify the properties of the medians and centroid:** - The medians of a triangle meet at the centroid P, which divides each median in a ratio of 2:1 (measured from the vertex to the midpoint). - Hence, $AP : PD = 2 : 1$ and $CP : PE = 2 : 1$.
2. **Determine the lengths of the segments:** - Given $PD = 2$, and using $PD = \frac{1}{3}AD$, we obtain $AD = 3 \times 2 = 6$. - Given $PE = 1.5$, and using $PE = \frac{1}{3}CE$, we obtain $CE = 3 \times 1.5 = 4.5$. - In addition, $DE = 2.5$. Since D and E are midpoints, DE is a midsegment, so $DE = \frac{1}{2}AC$, which implies $AC = 5$.
3. **Relate the medians to the area of the triangle:** - The three medians of a triangle form a new triangle whose area is $\frac{3}{4}$ of the area of the original triangle. - In this case, the medians have lengths 6, 4.5, and the third median can be found to be 7.5.
4. **Compute the area using the given lengths:** - Applying Heron's formula to the triangle formed by the medians: $s = \frac{6+4.5+7.5}{2} = 9$ Area $= \sqrt{9(9-6)(9-4.5)(9-7.5)} = 13.5$ - Therefore, the area of $\triangle ABC$ is: $[ABC] = \frac{4}{3} \times 13.5 = 18$
5. **Calculate the area of $AEDC$:** - Triangle BED is formed by connecting midpoints, so its area is $\frac{1}{4}$ of the area of $\triangle ABC$: $[BED] = \frac{1}{4} \times 18 = 4.5$ - Thus, the area of quadrilateral $AEDC$ is: $[AEDC] = [ABC] - [BED] = 18 - 4.5 = 13.5$
6. **Final answer:** - The area of $\triangle AEDC$ is $\boxed{13.5}$. ✔

Table 11: Case study of a visualized example from from validation set of Geometry3K with Qwen2.5-VL-7B. We highlight errors generated by Qwen2.5-VL-7B.



Question: In $\odot H$, $PQ = 3x - 4$ and $RS = 14$. Find x. Answer: 6

Qwen2.5-VL-7B

To solve for x in the given problem, we need to use the properties of circles and chords. Here's the step-by-step reasoning: 1. **Identify the given information:** - The circle is centered at H. - $PQ = 3x - 4$. - $RS = 14$. 2. **Understand the relationship between the chords and the diameter:** - Since T and U are points on the circle such that $TH \perp PQ$ and $UH \perp RS$, these perpendiculars bisect the chords PQ and RS respectively. - Therefore, $PT = TQ = \frac{PQ}{2}$ and $RU = US = \frac{RS}{2}$. 3. **Calculate the lengths of the segments:** - Given $PQ = 3x - 4$, the length of PT and TQ is:

$$PT = TQ = \frac{PQ}{2} = \frac{3x - 4}{2}$$

- Given $RS = 14$, the length of RU and US is:

$$RU = US = \frac{RS}{2} = \frac{14}{2} = 7$$

4. **Set up the equation using the total length of the chord PQ:** - Since $PQ = PT + TQ$:

$$PQ = \frac{3x - 4}{2} + \frac{3x - 4}{2} = 3x - 4$$

5. **Solve for x:** - We know $PQ = 3x - 4$ and from the diagram, $PQ = 16$ (since T and U are midpoints and the total length is twice the segment from the center to the midpoint of the chord). - Set up the equation:

$$3x - 4 = 16$$

- Solve for x:

$$3x = 20$$
$$x = \frac{20}{3}$$

Therefore, the value of x is $\boxed{\frac{20}{3}}$. ✘

NOPD

To solve for x in the given problem, we need to use the properties of circles and chords. Here's the step-by-step reasoning:
1. **Identify the given information:** - The circle is centered at H. - $PQ = 3x - 4$. - $RS = 14$. - The perpendicular distances from the center to both chords are equal: $HT = HU = 8$.
2. **Understand the relationship between the chords and the center:** - Since $HT \perp PQ$ and $HU \perp RS$, these perpendiculars bisect the chords PQ and RS respectively. - Therefore, $PT = TQ = \frac{PQ}{2}$ and $RU = US = \frac{RS}{2}$.
3. **Use the equal distance property of chords:** - In a circle, chords that are equidistant from the center are equal in length. - Since $HT = HU = 8$, chords PQ and RS are equal. - Hence, $PQ = RS$.
4. **Set up the equation:**

$$3x - 4 = 14$$

5. **Solve for x:**

$$3x = 18$$
$$x = 6$$

6. **Final answer:** - The value of x is $\boxed{6}$. ✔