MINERU-DIFFUSION V2: BRIDGING AUTOREGRESSIVE AND DIFFUSION DECODING FOR DOCUMENT OCR

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion decoding offers a new path to parallel generation for document OCR, but training from scratch leaves the capabilities of existing autoregressive parsers underused. We introduce MinerU-Diffusion v2, a 1.2B-parameter block diffusion document parser initialized from MinerU2.5-Pro. The design balances decoding parallelism and recognition quality while accounting for the downstream effects of layout decisions. Autoregressive-to-Diffusion Continued Training (ADCT) reuses existing document supervision through progressive block-size training and smaller-block replay, enabling a single checkpoint to support multiple inference block sizes. Hierarchical Joint Policy Optimization (HJPO) uses downstream rewards and conditional credit assignment to optimize final parsing quality, alternately updating a shared policy for layout generation and content recognition. On OmniDocBench v1.5, the model achieves Overall scores of 92.63 with predicted layouts and 94.82 with ground-truth layouts, outperforming standard MinerU2.5 in both settings. Compared with MinerU-Diffusion, it uses fewer than half the parameters while improving the respective scores by 3.69 and 1.45 points. Effective output throughput increases from 131.22 to 152.36 tokens per second, with flexible speed–accuracy trade-offs supported by configurable inference block sizes.

1 INTRODUCTION

Document optical character recognition (OCR) has expanded beyond text transcription to recovering formulas, tables, reading order, and layout structure from page images (Ouyang et al., 2025; Wang et al., 2024c). Two-stage parsers such as MinerU2.5 (Niu et al., 2025) first predict the page layout and then recognize content within the resulting regions. Autoregressive decoding generates output tokens sequentially, limiting decoding parallelism when processing long tables, formulas, and structured markup (Arriola et al., 2025). Improving decoding efficiency while preserving parsing quality is therefore a central challenge in document parsing.

Diffusion language models use bidirectional context to predict tokens in parallel (Nie et al., 2025; Ye et al., 2025). Block diffusion combines causal generation across blocks with parallel denoising within each block, allowing decoding parallelism to be adjusted flexibly (Arriola et al., 2025). MinerU-Diffusion (Dong et al., 2026) demonstrated the potential of this paradigm for document OCR, but its backbone was trained from scratch without initialization from an existing autoregressive parser. This motivates exploring how to convert a compact, task-aligned autoregressive parser into a block diffusion model with a modest additional training budget, preserving document recognition capabilities while supporting flexible inference configurations.

Beyond adapting the decoding objective, document parsing must address the dependence of content recognition on predicted layouts. Geometric overlap with annotated boxes does not fully capture a layout's value for downstream parsing. For example, a continuous paragraph can be represented as a single region or divided into several adjacent regions. These alternatives change the visual inputs and sequence boundaries presented to the recognition stage. Optimizing layout IoU alone therefore does not directly optimize final parsing quality. Moreover, page-level rewards reflect decisions from both layout generation and content recognition, motivating the evaluation of candidate layouts through

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.

downstream outcomes and credit assignment to recognition trajectories under their corresponding layout conditions.

We introduce MinerU-Diffusion v2, a 1.2B-parameter block diffusion document parser initialized from MinerU2.5-Pro (Wang et al., 2026a). Autoregressive-to-Diffusion Continued Training (ADCT) reuses existing document supervision through progressive block-size training and smaller-block replay, allowing a single checkpoint to support multiple inference block sizes. Block sizes and confidence thresholds can be configured for different document elements, providing flexible speed–accuracy trade-offs without retraining. ADCT also preserves much of the initializer's parsing performance: after two Stage-2 epochs, the model retains 98.6% of the initializer's Overall score on OmniDocBench v1.5. The conversion stage uses 7.8M sample exposures, equivalent to only 11.8% of the exposures in MinerU-Diffusion's reported training pipeline (Dong et al., 2026).

We further introduce Hierarchical Joint Policy Optimization (HJPO), which couples layout generation and conditional content recognition through a shared policy, using final page-level parsing quality as the common objective. HJPO compares mean downstream rewards across candidate layouts and recognition trajectories within the same layout condition, constructing credit assignment signals for the two stages. These signals guide alternating updates to the shared policy, aligning layout optimization directly with downstream parsing quality.

On OmniDocBench v1.5, V2 achieves Overall scores of 92.63 with predicted layouts and 94.82 with ground-truth layouts, outperforming standard MinerU2.5 in both settings. Compared with MinerU-Diffusion, it uses fewer than half the parameters and improves the respective scores by 3.69 and 1.45 points. Under the reported decoding settings, effective output throughput increases from 131.22 to 152.36 tokens per second. These results show that task-aligned autoregressive parsers provide a viable starting point for compact diffusion document parsers that combine strong parsing performance with adjustable parallelism.

2 Related Work

2.1 Document OCR

Modular document OCR pipelines (Smith, 2007; Wang et al., 2024c; Auer et al., 2024) coexist with end-to-end vision-language models that generate markup (Kim et al., 2022; Blecher et al., 2023; Lv et al., 2023). Two-stage parsers such as MinerU2.5 (Niu et al., 2025) perform low-resolution layout analysis followed by content recognition on high-resolution crops. MinerU-Diffusion (Dong et al., 2026) formulates document OCR as inverse rendering with parallel diffusion decoding. Our work converts an existing autoregressive document parser to block diffusion while retaining the two-stage architecture and jointly optimizing layout generation and content recognition.

2.2 Diffusion Language Models

Diffusion language models extend iterative denoising (Ho et al., 2020) to text through continuous token representations (Li et al., 2022) or discrete transition processes (Austin et al., 2021). Subsequent discrete formulations refine training objectives and parameterizations for efficient, stable learning (Lou et al., 2024; Sahoo et al., 2024; Ou et al., 2025; Shi et al., 2024). Block diffusion combines causal generation across blocks with bidirectional denoising within blocks, supporting variable-length generation and KV-cache reuse (Arriola et al., 2025). Dream initializes language diffusion from pretrained autoregressive weights (Ye et al., 2025), while Fast-dLLM v2 converts autoregressive language models to block diffusion (Wu et al., 2025). Fast-dVLM studies conversion of pretrained vision-language models, including block-size annealing (Wu et al., 2026). Within this line of work, ADCT converts an autoregressive document parser to block diffusion using its original task supervision, a block-size curriculum, and smaller-block replay (Section 3.1).

2.3 RL for Diffusion Language Models

Reinforcement learning methods such as PPO (Schulman et al., 2017) and GRPO (Shao et al., 2024) optimize language models using reward signals. For diffusion models, multiple denoising steps and masked-token likelihood estimation complicate policy optimization. MMaDA estimates objectives

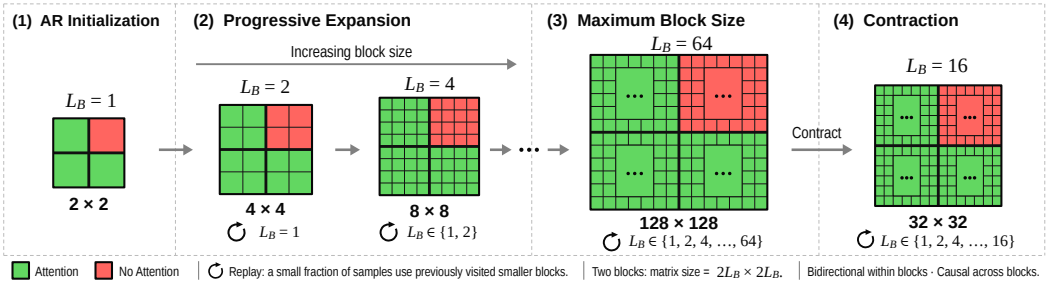


Figure 1: ADCT overview. Starting from autoregressive decoding ($L_B = 1$), training progressively expands the block size to 64 and then contracts it to 16, with smaller-block replay. Attention masks are visualized using two consecutive blocks.

using random masks (Yang et al., 2025a), while DiffuCoder uses complementary masks to reduce estimation variance (Gong et al., 2025). LLaDA 1.5 applies preference optimization (Zhu et al., 2025). TraceRL (Wang et al., 2026b) aligns optimization with inference traces via a step-wise MDP. In document parsing, recognition outcomes also depend on the layout that determines the recognition inputs. HJPO applies trajectory-based policy optimization to these coupled stages of layout generation and content recognition. It compares layouts using mean downstream parsing rewards and compares recognition trajectories conditioned on the same layout. These conditional comparisons guide alternating updates to a shared policy. Appendix B further discusses speculative OCR decoding, AR-to-diffusion conversion, and document-level reward optimization.

3 Methodology

3.1 Autoregressive-to-Diffusion Continued Training

Autoregressive-to-Diffusion Continued Training (ADCT) converts MinerU2.5-Pro using its original supervision, preserving its visual architecture, tokenizer, prompts, and output representations (Wang et al., 2026a). Figure 1 shows the block-size curriculum and replay.

Let $\mathbf{x}$ denote a page image or region crop, q the task prompt, and $\mathbf{y}_0 = (y_1, \ldots, y_L)$ the target sequence. Targets comprise plain text, LaTeX expressions, OTSL (Lysak et al., 2023) table sequences, or layout tuples. A block size L_B partitions the target into $B(L_B) = \lceil L/L_B \rceil$ contiguous blocks, generated causally with bidirectional masked denoising within each block (Arriola et al., 2025). Blocks condition on the image, prompt, and completed prefix; $L_B = 1$ recovers token-wise autoregressive decoding, the starting point for conversion.

Progressive block-size training. The primary block size grows from $L_B = 1$ to 64, then contracts to 16 for continued training. During expansion and at $L_B = 64$, most samples use the primary size, while a small fraction replay previously visited smaller sizes. Contraction follows directly, without a separate stable-training or learning-rate annealing stage; smaller-block replay continues at $L_B = 16$.

At training stage s, we sample $L_B \sim \pi_s$ from the stage-specific block-size distribution, a diffusion timestep t, and $(\mathbf{x}, q, \mathbf{y}_0) \sim \mathcal{D}_{\text{doc}}$. The corruption process Q_t produces $\mathbf{y}_t$ by replacing each target token with `[MASK]` with probability $1 - \alpha_t$. Following the block-diffusion objective (Arriola et al., 2025), we minimize

$$\mathcal{L}_{\text{ADCT}}(\theta) = \mathbb{E}\left[w(t) \sum_{b=1}^{B(L_B)} \sum_{i \in \mathcal{M}(\mathbf{y}_{t,b})} -\log p_\theta\left(y^i_{0,b} \mid \mathbf{x}, q, \mathbf{y}_{0,<b}, \mathbf{y}_{t,b}\right) \right], \tag{1}$$

where $w(t) = -\alpha'_t/(1 - \alpha_t)$ is nonnegative for a decreasing retention schedule α_t, and $\mathcal{M}(\mathbf{y}_{t,b})$ denotes masked positions in block b.

Adaptive block-size decoding. Progressive training and smaller-block replay enable multiple inference block sizes in one checkpoint: smaller blocks ease reconstruction, while larger blocks in-

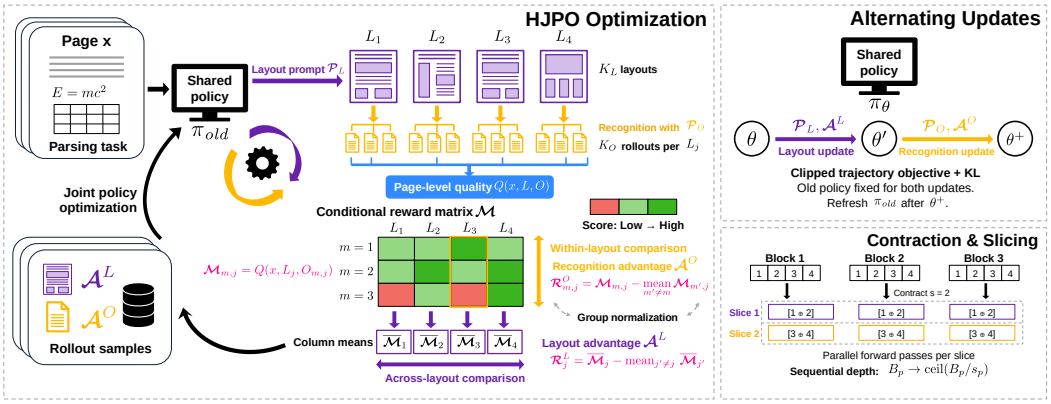


Figure 2: HJPO overview. Left: hierarchical rollouts and conditional credit assignment. Top right: alternating updates of a shared policy. Bottom right: slice-wise training for layout generation and content recognition.

crease parallelism. Element-specific block sizes and confidence thresholds adjust this trade-off without retraining.

3.2 Hierarchical Joint Policy Optimization

Hierarchical Joint Policy Optimization (HJPO) couples layout generation and recognition through final page-level parsing quality. It compares layout candidates and recognition trajectories conditioned on each layout. A shared policy π_θ uses prompts P_L and P_O for the two stages. Joint optimization denotes the common objective and conditional rewards; alternating optimization denotes sequential updates of the shared parameters. Adapting TraceRL (Wang et al., 2026b), trajectory contraction groups diffusion steps into segments, and slice-wise training batches corresponding positions across blocks to reduce sequential policy forward passes. Figure 2 summarizes the resulting training workflow.

Hierarchical rollouts and conditional rewards. A paragraph may be represented as one region or several adjacent regions; we optimize their downstream parsing utility directly:

$$\max_{\theta} \ \mathbb{E}_{x\sim\mathcal{D}}\left[Q(x,L,O)\right], \qquad L\sim\pi_\theta(\cdot\mid x,P_L), \quad O\sim\pi_\theta(\cdot\mid x,L,P_O), \tag{2}$$

where $Q(x, L, O)$ scores layout L and parsing result O. We compare layouts and recognition choices separately. For each page, $\pi_{\theta_{\text{old}}}$ samples K_L layout trajectories τ_j^L yielding L_j and K_O recognition trajectories $\tau_{m,j}^O$ yielding $O_{m,j}$ per layout. Their conditional reward matrix and layout means are

$$M\in\mathbb{R}^{K_O\times K_L}, \qquad M_{m,j}=Q(x,L_j,O_{m,j}), \qquad \bar{M}_j=\frac{1}{K_O}\sum_{m=1}^{K_O}M_{m,j}. \tag{3}$$

The mean $\bar{M}_j$ estimates layout L_j's expected downstream utility under the old recognition policy. Recognition trajectories are compared only within the same layout condition. Building on baseline control variates (Greensmith et al., 2004), we define nested leave-one-out rewards for $K_L, K_O > 1$:

$$R_j^L=\bar{M}_j-\frac{1}{K_L-1}\sum_{\substack{j'=1\\ j'\neq j}}^{K_L}\bar{M}_{j'}, \qquad R_{m,j}^O=M_{m,j}-\frac{1}{K_O-1}\sum_{\substack{m'=1\\ m'\neq m}}^{K_O}M_{m',j}. \tag{4}$$

Layout rewards compare candidates for the same page; recognition rewards compare trajectories within a layout. This assigns conditional credit without strict causal separation of errors. Normalizing across layouts and within each layout condition gives A_j^L and $A_{m,j}^O$, respectively.

Prompt-conditioned alternating policy optimization. For prompt P, let $\mathcal{T} = \{(c_i, \tau_i, A_i)\}_{i=1}^{G}$ be a rollout set collected by the old policy. Here, c_i is the input condition, $\tau_i = (a_i(1), \ldots, a_i(T_i))$ the trajectory, and A_i its normalized advantage. Following TraceRL's contraction strategy (Wang et al., 2026b), we group every s consecutive diffusion steps into an ordered segment:

$$z_{i,t}^{s} = \bigoplus_{h=s(t-1)+1}^{\min(st,T_i)} a_i(h), \qquad T_i^s = \left\lceil \frac{T_i}{s} \right\rceil, \tag{5}$$

where $\oplus$ denotes ordered concatenation. Segment probabilities use the underlying diffusion model's within-segment factorization. With history $h_{i,t}^{s} = (c_i, P, z_{i,1:t-1}^{s})$, the importance ratio is

$$r_{i,t}(\theta) = \frac{\pi_\theta(z_{i,t}^{s} \mid h_{i,t}^{s})}{\pi_{\theta_{\text{old}}}(z_{i,t}^{s} \mid h_{i,t}^{s})}. \tag{6}$$

Using the PPO clipped surrogate (Schulman et al., 2017)

$$C_\epsilon(r, A) = \min\left(rA,\ \text{clip}(r, 1-\epsilon, 1+\epsilon)A\right), \tag{7}$$

we optimize the common contracted-trajectory objective

$$\begin{aligned} \mathcal{J}^{s}(\theta; P, \mathcal{T}) = \mathbb{E}_{\mathcal{T}}\Bigg[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{T_i^s} \sum_{t=1}^{T_i^s} \Bigg(& C_\epsilon(r_{i,t}(\theta), A_i) \\ & - \beta D_{\text{KL}}\left(\pi_\theta(\cdot \mid h_{i,t}^{s}) \,\|\, \pi_{\theta_{\text{old}}}(\cdot \mid h_{i,t}^{s})\right) \Bigg) \Bigg]. \end{aligned} \tag{8}$$

The two rollout sets specialize this objective to layout generation and content recognition:

$$\begin{aligned} \mathcal{T}_L &= \{(x, \tau_j^L, A_j^L)\}_{j=1}^{K_L}, \\ \mathcal{T}_O &= \{((x, L_j), \tau_{m,j}^O, A_{m,j}^O)\}_{1 \le j \le K_L,\, 1 \le m \le K_O}. \end{aligned} \tag{9}$$

Their objectives are applied sequentially to the shared parameters:

$$\begin{aligned} \theta' &= \theta + \eta_L \nabla_\theta \mathcal{J}^{s_L}(\theta; P_L, \mathcal{T}_L), \\ \theta^+ &= \theta' + \eta_O \nabla_{\theta'} \mathcal{J}^{s_O}(\theta'; P_O, \mathcal{T}_O). \end{aligned} \tag{10}$$

The old policy remains fixed during both updates and is refreshed with θ^+ only for the next rollout iteration. As in GRPO (Shao et al., 2024), group-relative advantages remove the need for an auxiliary value function or critic.

Slice-wise training for layout generation and content recognition. For task $p \in \{L, O\}$, let block r of trajectory i contain $B_{i,r}^{p} \le B_p$ diffusion steps. Within-block contraction by s_p yields

$$S_{i,r}^{p} = \left\lceil \frac{B_{i,r}^{p}}{s_p} \right\rceil, \qquad S_{\max}^{p} = \max_{i,r} S_{i,r}^{p} \le \left\lceil \frac{B_p}{s_p} \right\rceil. \tag{11}$$

Writing $z_{i,r,l}^{p}$ for the l-th contracted segment, we adapt TraceRL's slice-wise training (Wang et al., 2026b) by re-indexing policy terms across blocks:

$$\sum_{r} \sum_{l=1}^{S_{i,r}^{p}} f(z_{i,r,l}^{p}) = \sum_{l=1}^{S_{\max}^{p}} \sum_{r} f(z_{i,r,l}^{p}) \mathbf{1}\{l \le S_{i,r}^{p}\}. \tag{12}$$

Under a block-causal attention mask, valid segments at the same relative position form one slice and undergo parallel forward computation. The sequential depth decreases from at most B_p to $\lceil B_p / s_p \rceil$ for each task, independently of the number of blocks under ideal batching. The theoretical derivations are provided in Appendix A.

Table 1: Comprehensive document parsing results on OmniDocBench v1.5. ↑ denotes higher is better, ↓ denotes lower is better, and dLM denotes diffusion language model decoding. Shaded rows are ours.

Type	Method	Params.	Overall ↑	Text ↓	Formula ↑	Tab_{TEDS} ↑	$\text{Tab}_{\text{TEDS-S}}$ ↑	Read. Ord. ↓	GT Layout
Pipeline	MinerU2-pipeline (Wang et al., 2024c)	–	75.51	0.209	76.55	70.90	79.11	0.225	✗
	PP-StructureV3 (Cui et al., 2025b)	–	86.73	0.073	85.79	81.68	89.48	0.073	✗
AR	Qwen2.5-VL-72B (Bai et al., 2025)	72B	87.02	0.094	88.27	82.15	86.22	0.102	✗
	Gemini-2.5 Pro (Comanici et al., 2025)	–	88.03	0.075	85.82	85.71	90.29	0.097	✗
	MinerU2-VLM (Wang et al., 2024c)	0.9B	85.56	0.078	80.95	83.54	87.66	0.086	✗
	Nanonets-OCR-s (Mandal et al., 2025)	3B	85.59	0.093	85.90	80.14	85.57	0.108	✗
	MonkeyOCR-pro-1.2B (Li et al., 2025b)	1.9B	86.96	0.084	85.02	84.24	89.02	0.130	✗
	MonkeyOCR-pro-3B (Li et al., 2025b)	3.7B	88.85	0.075	87.25	86.78	90.63	0.128	✗
	dots.ocr (Li et al., 2025a)	3B	88.41	0.048	83.22	86.78	90.62	0.053	✗
	MinerU2.5 (Niu et al., 2025)	1.2B	90.67	0.047	88.46	88.22	92.38	0.044	✗
	PaddleOCR-VL (Cui et al., 2025a)	0.9B	92.56	0.035	91.43	89.76	93.52	0.043	✗
dLM	MinerU-Diffusion (Dong et al., 2026)	2.5B	88.94	0.061	86.41	86.50	90.29	0.059	✗
dLM	MinerU-Diffusion v2	1.2B	92.63	0.037	92.69	88.90	92.80	0.045	✗
AR	MinerU2.5 (Niu et al., 2025)	1.2B	93.44	0.025	91.98	90.84	95.10	–	✓
	PaddleOCR-VL (Cui et al., 2025a)	0.9B	93.91	0.021	92.13	91.70	95.42	–	✓
	MinerU2.5-Pro (Wang et al., 2026a)	1.2B	95.39	0.016	93.60	94.18	97.24	–	✓
dLM	MinerU-Diffusion	2.5B	93.37	0.028	91.92	91.00	94.86	–	✓
dLM	MinerU-Diffusion v2	1.2B	94.82	0.022	94.10	92.56	96.62	–	✓

4 Experiments

4.1 Setup

4.1.1 Data

ADCT reuses the 3.9M deduplicated Stage-2 samples from MinerU2.5-Pro (Wang et al., 2026a), covering layout, OCR, and image analysis, including 192,000 manually annotated samples. HJPO uses MinerU2.5-Pro's Stage-3 training data. No new data are collected or synthesized; the larger-recipe ablation adds Stage-1 data beyond the replay already included in the Stage-2 mixture. The corpus focuses on Chinese and English; dedicated low-resource-language evaluation is outside this study.

4.1.2 Model and Optimization

We initialize MinerU-Diffusion v2 from the 1.2B-parameter autoregressive MinerU2.5-Pro. The primary ADCT configuration uses two Stage-2 epochs (7.8M sample exposures), and a maximum sequence length of 16,384 tokens. This conversion budget excludes autoregressive pretraining and uses no additional VQA warm-up or post-CPT supervised fine-tuning. It corresponds to 11.8% of the 65.909M exposures derived from the reported Stage-0–2 sample counts and epochs of MinerU-Diffusion (Dong et al., 2026, Table 6), excluding inherited backbone pretraining. Appendix E.1 summarizes the training configurations; Appendix E.2 details the ADCT data mixture and optimization settings.

4.1.3 Evaluation

We evaluate full-document parsing and layout analysis on OmniDocBench v1.5 (Ouyang et al., 2025). Table recognition uses CC-OCR (Yang et al., 2025b) and OCRBench v2 (Fu et al., 2025), while formula recognition uses UniMER-Test (Wang et al., 2024a), with task-specific prompts throughout (Appendix D). TextEdit and Reading Order edit distance are lower-is-better; CDM (Wang et al., 2024b), TEDS (Zhong et al., 2020), TEDS-S, and Overall are higher-is-better, following the OmniDocBench v1.5 protocol. We test predicted layouts for end-to-end parsing and ground-truth layouts for content recognition. Baselines span pipeline systems, general-purpose vision-language models, autoregressive MinerU2.5 (Niu et al., 2025), and MinerU-Diffusion. Appendix E.4 specifies metric aggregation, decoding configurations, and evaluation hardware.

Table 2: Recognition results on CC-OCR, OCRBench v2, and UniMER-Test. ↑ denotes higher is better. † denotes results reported by MinerU2.5-Pro (Wang et al., 2026a) under updated evaluation versions.

		CC-OCR		OCRBench v2		UniMER-Test			
Type	Method	TEDS	TEDS-S	TEDS	TEDS-S	CPE	HWE	SCE	SPE
AR	InternVL3.5-241B (Wang et al., 2025)	62.87	69.52	79.50	85.81	91.7	93.2	95.1	97.8
	Qwen2.5-VL-72B (Bai et al., 2025)	81.22	86.48	81.33	86.58	88.9	91.8	95.5	96.2
	GPT-4o (OpenAI, 2024)	66.98	79.04	70.51	79.55	82.7	85.9	87.8	96.7
	dots.ocr (Li et al., 2025a)	75.42	81.65	82.04	86.27	86.8	90.5	94.7	97.5
	MinerU2.5 (Niu et al., 2025)	79.76	85.16	87.13	90.62	96.6	94.4	96.4	98.4
	MinerU2.5-Pro† (Wang et al., 2026a)	88.49	91.90	93.56	96.27	99.0	95.4	97.0	99.4
dLM	MinerU-Diffusion (Dong et al., 2026)	73.77	82.06	81.18	88.66	91.6	91.6	92.0	96.8
dLM	MinerU-Diffusion v2	83.54	89.21	88.28	92.36	96.7	94.6	95.3	98.2

4.2 Main Results

4.2.1 Full-document Parsing

With predicted layouts, MinerU-Diffusion v2 reaches 92.63 Overall, exceeding MinerU-Diffusion by 3.69 points with fewer than half the parameters (1.2B versus 2.5B; Table 1). It also exceeds all autoregressive baselines reported under predicted layouts in this table. With ground-truth (GT) layouts, it reaches 94.82, exceeding MinerU-Diffusion, MinerU2.5, and PaddleOCR-VL but remaining below MinerU2.5-Pro (95.39). Its GT–Pred gap is 2.19 versus 4.43 for MinerU-Diffusion, a 50.6% reduction; Section 4.3 isolates the effects of conversion and policy optimization.

4.2.2 Element-specific Parsing

Table recognition. Relative to MinerU-Diffusion, MinerU-Diffusion v2 improves CC-OCR TEDS/TEDS-S by 9.77/7.15 points and OCRBench v2 TEDS/TEDS-S by 7.10/3.70 points (Table 2). MinerU2.5-Pro scores higher under updated evaluation versions.

Formula recognition. On UniMER-Test, MinerU-Diffusion v2 improves CPE, HWE, SCE, and SPE over MinerU-Diffusion by 5.1, 3.0, 3.3, and 1.4 points, respectively. MinerU2.5-Pro remains higher across all four categories.

4.2.3 Inference Speed

At its quality-oriented operating point, MinerU-Diffusion v2 achieves 94.82 Overall, 4.45 tokens committed per forward pass (TPF), and 152.36 effective output tokens/s (TPS; Table 3). Despite lower TPF than MinerU-Diffusion (5.04), throughput rises from 131.22 to 152.36 TPS with the smaller backbone.

Table 3: Speed–accuracy comparison of recent OCR models on OmniDocBench v1.5. TPF denotes tokens committed per forward pass, and TPS denotes effective output tokens per second. * denotes official decoding results used when ground-truth-layout decoding could not be run.

Type	Methods	Params.	TPF ↑	Score ↑	TPS ↑
AR	Qwen3.5 (Qwen Team, 2026)	4B	1.00	83.45	32.30
	olmOCR 2* (Poznanski et al., 2025)	7B	1.00	84.44	51.50
	DeepSeek-OCR 2* (Wei et al., 2026)	3B	1.00	90.64	57.58
	MinerU2.5 (Niu et al., 2025)	1.2B	1.00	93.44	51.46
	PaddleOCR-VL (Cui et al., 2025a)	0.9B	1.00	93.91	40.77
	GLM-OCR (Duan et al., 2026)	0.9B	1.00	95.27	77.99
	MinerU2.5-Pro (Wang et al., 2026a)	1.2B	1.00	95.39	51.46
dLM	MinerU-Diffusion (Dong et al., 2026)	2.5B	5.04	93.37	131.22
dLM	MinerU-Diffusion v2	1.2B	4.45	94.82	152.36

4.3 Ablations

4.3.1 Conversion Efficiency and Inference Block Length

We examine the effects of ADCT training duration, recipe, and initialization at threshold 0.95 and uniform block lengths $L' \in \{8, 16, 32, 64\}$ (Table 4). Following the Stage-1/Stage-2 definitions of MinerU2.5-Pro (Wang et al., 2026a), the compared configurations comprise one to three epochs of Stage-2 training and two epochs of Stage-2 training with an additional 65.5M Stage-1 samples.

Table 4: Training duration, recipe, and initialization ablations. S1 and S2 denote Stage-1 and Stage-2, respectively. Each block-length column reports Overall / TPF. Retention is the dLM Overall at $L' = 8$ divided by the autoregressive Overall of the initialization.

Training recipe	Initialization (AR Overall)	Sample exposures	Retention	$L' = 8$	$L' = 16$	$L' = 32$	$L' = 64$
S2 × 1 epoch	MinerU2.5-Pro (95.39)	3.9M	98.1%	93.59 / 3.36	90.93 / 3.84	88.46 / 3.99	85.50 / 3.44
S2 × 2 epochs	MinerU2.5-Pro (95.39)	7.8M	98.6%	94.02 / 4.06	93.39 / 4.86	90.51 / 5.10	89.88 / 4.43
S2 × 3 epochs	MinerU2.5-Pro (95.39)	11.7M	98.1%	93.57 / 3.76	92.68 / 4.48	90.40 / 4.72	89.23 / 4.06
S1 + S2 × 2 epochs	MinerU2.5-Pro (95.39)	65.5M + 7.8M	98.4%	93.85 / 4.12	93.63 / 5.42	91.95 / 6.69	89.39 / 7.26
S2 × 2 epochs	MinerU2.5 (93.44)	7.8M	98.5%	92.05 / 3.29	92.55 / 4.26	89.24 / 5.14	87.95 / 5.62

One epoch establishes competitive diffusion decoding. One pass over 3.9M Stage-2 samples yields 93.59 Overall at $L' = 8$, retaining 98.1% of MinerU2.5-Pro's score and exceeding the larger MinerU-Diffusion model (93.37).

Two epochs favor accuracy; Stage-1 data improve large-block parallelism. Two Stage-2 epochs give the highest accuracy among the Stage-2-only checkpoints, improving Overall by 0.43–4.38 points and TPF by 20.8%–28.8% over one epoch. A third epoch provides no further gain across the evaluated block lengths. Adding Stage-1 data does not consistently improve accuracy, but increases parallelism most at larger blocks. At $L' = 64$, TPF rises from 4.43 to 7.26 (+63.9%), while Overall falls by 0.49 points.

Initialization quality carries over to the converted model. With the same 7.8M sample exposures, the MinerU2.5 and MinerU2.5-Pro initializations retain 98.5% and 98.6% of their autoregressive performance at $L' = 8$, respectively. MinerU2.5-Pro initialization also outperforms MinerU2.5 initialization by 0.84–1.97 points across all inference block lengths.

4.3.2 Inference-Time Element-Aware Block Allocation

As described in Section 3.1, progressive block-size training and smaller-block replay in ADCT enable a single checkpoint to support multiple inference block sizes without retraining. Building on this capability, we allocate decoding block sizes according to document element types to balance parsing quality and parallelism. Table 5 compares uniform block lengths with element-aware block allocation under ground-truth layouts. Uniform configurations use a confidence threshold of 0.95; element-aware configurations use 0.92/0.96/0.92 for formula/table/text. Within the element-aware configurations, thresholds remain fixed while text and table block lengths vary separately.

Table 5: Element-aware block allocation on OmniDocBench v1.5 under ground-truth layouts. Block lengths and element-specific thresholds follow formula/table/text order.

Setting	Block lengths	Thresholds	Overall ↑	Text ↓	Formula ↑	Tab_{TEDS} ↑	TPF ↑
Uniform	8/8/8	0.95	94.02	0.0295	93.08	91.93	4.06
Uniform	16/16/16	0.95	93.39	0.0415	92.92	91.40	4.86
Uniform	32/32/32	0.95	90.51	0.0779	90.66	88.66	5.10
Element-aware	4/8/8	0.92/0.96/0.92	94.32	0.0292	93.84	92.04	4.45
Element-aware	4/8/16	0.92/0.96/0.92	94.10	0.0333	93.72	91.91	5.60
Element-aware	4/16/8	0.92/0.96/0.92	94.15	0.0272	93.98	91.20	4.55

Element-aware allocation provides greater inference flexibility. Element-specific block lengths and thresholds offer speed–accuracy trade-offs without retraining. Settings 4/8/8 and 4/8/16 achieve

94.32 and 94.10 Overall at 4.45 and 5.60 TPF, respectively, exceeding uniform lengths 8 and 16 by 0.30 and 0.71 Overall points while also increasing TPF.

4.3.3 Downstream Coupling and Conditional Credit Assignment

We initialize HJPO from MinerU2.5-Pro (AR Overall 95.39) after two S2 epochs of ADCT. Appendix E.3 details the three RL configurations' inputs, rewards, rollout counts, and normalization.

Page-level downstream coupling aligns local optimization with final parsing quality. Independent GRPO (Shao et al., 2024) optimizes layout IoU and recognition metrics separately, yet reduces Pred-layout Overall from ADCT's 91.81 to 91.13 (Table 6). A shared page-level reward improves Pred-layout and GT-layout Overall by 0.91 and 0.36 points, respectively, narrowing the GT–Pred gap from 2.95 to 2.40. Joint optimization evaluates alternative region partitions by downstream parsing utility, which geometric agreement alone does not capture (Section 3.2).

Conditional credit assignment provides stage-specific signals for the shared objective. Conditional comparisons further improve GT-layout and Pred-layout Overall by 0.38 and 0.59 points to 94.82 and 92.63, respectively, and reduce Reading Order edit distance from 0.048 to 0.045. These gains complement downstream coupling.

Table 6: HJPO ablation results on OmniDocBench v1.5. GT-layout and Pred-layout use ground-truth and predicted layouts, respectively; Gap is GT minus Pred Overall. Additions are cumulative, starting from GRPO.

Method	GT-layout Overall ↑	Pred-layout Overall ↑	Gap ↓	Reading Order ↓
MinerU2.5-Pro-2605-1.2B	95.39	93.07	2.32	0.038
MinerU-Diffusion	93.37	88.94	4.43	0.059
ADCT	94.32	91.81	2.51	0.053
GRPO	94.08	91.13	2.95	0.062
+ Downstream coupling	94.44	92.04	2.40	0.048
+ Conditional credit assignment (HJPO)	94.82	92.63	2.19	0.045

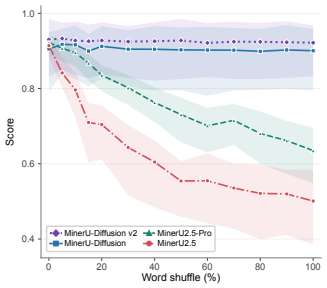


Figure 3: Semantic Shuffle. Lines show six-metric means; bands show metricwise min–max ranges.

4.4 Semantic Shuffle Analysis

We test whether autoregressive initialization carries over sensitivity to semantic context on Semantic Shuffle (Dong et al., 2026) (Figure 3), which perturbs word order in FOX document images (Liu et al., 2024) while keeping rendering format consistent. Despite initialization from MinerU2.5-Pro, MinerU-Diffusion v2 follows the stable perturbation trend of the diffusion baseline MinerU-Diffusion. Their mean absolute endpoint changes across six metrics are 0.009 and 0.012, respectively, versus 0.287 for MinerU2.5-Pro. On this benchmark, the converted model therefore does not retain its initializer's strong sensitivity to disrupted semantic coherence. Appendix C.1 provides the perturbation setup and six metric curves.

5 Conclusion

We presented MinerU-Diffusion v2, a compact document parser that converts task-aligned autoregressive capabilities into block diffusion decoding. ADCT reuses existing supervision to retain much of the initializer's parsing performance while supporting flexible inference parallelism. HJPO couples layout generation and conditional recognition through downstream rewards and hierarchical credit assignment. On OmniDocBench v1.5, V2 improves accuracy and reported throughput over standard MinerU2.5 and MinerU-Diffusion, using fewer than half the parameters of the latter. These results support a practical route to efficient diffusion OCR, while the remaining accuracy gap to MinerU2.5-Pro motivates better capability retention.

USE OF LARGE LANGUAGE MODELS (LLMS)

We used an AI assistant solely to polish the writing of this manuscript.

REFERENCES

Marianne Arriola, Aaron Gokaslan, Justin T Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *International Conference on Learning Representations (ICLR)*, 2025.

Christoph Auer, Maksym Lysak, Ahmed Nassar, Michele Dolfi, Nikolaos Livathinos, Panos Vagenas, Cesar Berrospi Ramis, Matteo Omenetti, Fabian Lindlbauer, Kasper Dinkla, et al. Docling technical report. *arXiv preprint arXiv:2408.09869*, 2024.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, pp. 17981–17993, 2021. URL https://papers.nips.cc/paper_files/paper/2021/hash/958c530554f78bcd8e97125b70e6973d-Abstract.html.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Alejandro Barón García, Feng Wang, Emilia Garcia Casademont, and Han Xiao. Jina-OCR-v1: Efficient document parsing with speculative decoding and dense verifiable rewards. *arXiv preprint arXiv:2609.03181*, 2026. URL https://arxiv.org/abs/2609.03181.

Lukas Blecher, Guillem Cucurull, Thomas Scialom, and Robert Stojnic. Nougat: Neural optical understanding for academic documents. *arXiv preprint arXiv:2308.13418*, 2023.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Cheng Cui, Ting Sun, Suyin Liang, Tingquan Gao, Zelun Zhang, Jiaxuan Liu, Xueqing Wang, Changda Zhou, Hongen Liu, Manhui Lin, Yue Zhang, Yubo Zhang, Handong Zheng, Jing Zhang, Jun Zhang, Yi Liu, Dianhai Yu, and Yanjun Ma. PaddleOCR-VL: Boosting multilingual document parsing via a 0.9b ultra-compact vision-language model. *arXiv preprint arXiv:2510.14528*, 2025a.

Cheng Cui, Ting Sun, Manhui Lin, Tingquan Gao, Yubo Zhang, Jiaxuan Liu, Xueqing Wang, Zelun Zhang, Changda Zhou, Hongen Liu, et al. PaddleOCR 3.0 technical report. *arXiv preprint arXiv:2507.05595*, 2025b. URL https://arxiv.org/abs/2507.05595.

Hejun Dong, Junbo Niu, Bin Wang, Weijun Zeng, Wentao Zhang, and Conghui He. MinerU-Diffusion: Rethinking document OCR as inverse rendering via diffusion decoding. *arXiv preprint arXiv:2603.22458*, 2026.

Shuaiqi Duan, Yadong Xue, Weihan Wang, Zhe Su, Huan Liu, Sheng Yang, Guobing Gan, Guo Wang, Zihan Wang, Shengdong Yan, Dexin Jin, Yuxuan Zhang, Guohong Wen, Yanfeng Wang, Yutao Zhang, Xiaohan Zhang, Wenyi Hong, Yukuo Cen, Da Yin, Bin Chen, Wenmeng Yu, Xiaotao Gu, and Jie Tang. GLM-OCR technical report, 2026. URL https://arxiv.org/abs/2603.10910.

Ling Fu, Zhebin Kuang, Jiajun Song, Mingxin Huang, Biao Yang, Yuzhe Li, Linghao Zhu, Qidi Luo, Xinyu Wang, Hao Lu, et al. OCRBench v2: An improved benchmark for evaluating large multimodal models on visual text localization and reasoning. In *Advances in Neural Information Processing Systems*, volume 38, pp. 107930–107981, 2025. doi: 10.52202/085713-3253.

Shansan Gong, Ruixiang Zhang, Huangjie Zheng, Jiatao Gu, Navdeep Jaitly, Lingpeng Kong, and Yizhe Zhang. DiffuCoder: Understanding and improving masked diffusion models for code generation. *arXiv preprint arXiv:2506.20639*, 2025.

Evan Greensmith, Peter L. Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5:1471–1530, 2004. URL https://www.jmlr.org/papers/v5/greensmith04a.html.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Dohyun Kim, Sungjun Han, Hyungguk Kim, Yusik Kim, Jamin Shin, Paul Hongsuck Seo, and Hongjoon Ahn. Diffusion drafts, AR verifies: Accelerating document OCR with self-speculative decoding. *arXiv preprint arXiv:2609.26638*, 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.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19274–19286, 2023. URL https://proceedings.mlr.press/v202/leviathan23a.html.

Gengluo Li, Xingyu Wan, Shangpin Peng, Weinong Wang, Hao Feng, Yongkun Du, Binghong Wu, Zheng Ruan, Zhiqiong Lu, Liang Wu, Pengyuan Lyu, Huawen Shen, Zibin Lin, Shijing Hu, Jieneng Yang, Hongbing Wen, Guanghua Yu, Hong Liu, Bochao Wang, Can Ma, Han Hu, Chengquan Zhang, and Yu Zhou. HunyuanOCR-1.5: Making lightweight OCR VLMs faster and better. *arXiv preprint arXiv:2607.04884*, 2026.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-LM improves controllable text generation. In *Advances in Neural Information Processing Systems*, volume 35, pp. 4328–4343, 2022. doi: 10.52202/068431-0313. URL https://papers.nips.cc/paper_files/paper/2022/hash/1be5bc25d50895ee656b8c2d9eb89d6a-Abstract-Conference.html.

Yumeng Li, Guang Yang, Hao Liu, Bowen Wang, and Colin Zhang. dots.ocr: Multilingual document layout parsing in a single vision-language model, 2025a. URL https://arxiv.org/abs/2512.02498.

Zhang Li, Yuliang Liu, Qiang Liu, Zhiyin Ma, Ziyang Zhang, Shuo Zhang, Biao Yang, Zidun Guo, Jiarui Zhang, Xinyu Wang, and Xiang Bai. MonkeyOCR: Document parsing with a structure-recognition-relation triplet paradigm. *arXiv preprint arXiv:2506.05218*, 2025b.

Wenhui Liao, Hongliang Li, Pengyu Xie, Xinyu Cai, Yufan Shen, Yi Xin, Qi Qin, Shenglong Ye, Tianbin Li, Ming Hu, Junjun He, Yihao Liu, Wenhai Wang, Min Dou, Bin Fu, Botian Shi, Yu Qiao, and Lianwen Jin. HSD: Training-free acceleration for document parsing vision-language models with hierarchical speculative decoding. *arXiv preprint arXiv:2602.12957v3*, 2026. URL https://arxiv.org/abs/2602.12957v3.

Chenglong Liu, Haoran Wei, Jinyue Chen, Lingyu Kong, Zheng Ge, Zining Zhu, Liang Zhao, Jianjian Sun, Chunrui Han, and Xiangyu Zhang. Focus anywhere for fine-grained multi-page document understanding. *arXiv preprint arXiv:2405.14295*, 2024.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *International Conference on Machine Learning (ICML)*, 2024.

Tengchao Lv, Yupan Huang, Jingye Chen, Lei Cui, Shuming Ma, Yaoyao Chang, Shaohan Huang, Wenhui Wang, Li Dong, Weiyao Luo, et al. Kosmos-2.5: A multimodal literate model. *arXiv preprint arXiv:2309.11419*, 2023.

Maksym Lysak, Ahmed Nassar, Nikolaos Livathinos, Christoph Auer, and Peter Staar. Optimized table tokenization for table structure recognition. In *Document Analysis and Recognition - ICDAR 2023*, pp. 37–50. Springer Nature Switzerland, 2023. doi: 10.1007/978-3-031-41679-8_3. URL https://doi.org/10.1007/978-3-031-41679-8_3.

Sean Man, Gilad Deutch, Roy Ganz, Roi Ronen, Shahar Tsiper, Shai Mazor, and Niv Nayman. DODO: Discrete OCR diffusion models. *arXiv preprint arXiv:2602.16872*, 2026.

Souvik Mandal, Ashish Talewar, Paras Ahuja, and Prathamesh Juvatkar. Nanonets-OCR-S: A model for transforming documents into structured markdown with intelligent content recognition and semantic tagging, 2025. URL https://huggingface.co/nanonets/Nanonets-OCR-s. Hugging Face model card.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025.

Junbo Niu, Zheng Liu, Zhuangcheng Gu, Bin Wang, Linke Ouyang, Zhiyuan Zhao, Tao Chu, Tianyao He, Fan Wu, Qintong Zhang, Zhenjiang Jin, et al. MinerU2.5: A decoupled vision-language model for efficient high-resolution document parsing. *arXiv preprint arXiv:2509.22186*, 2025.

OpenAI. GPT-4o system card, August 2024. URL https://cdn.openai.com/gpt-4o-system-card.pdf.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. *International Conference on Learning Representations (ICLR)*, 2025.

Linke Ouyang, Yuan Qu, Hongbin Zhou, Jiawei Zhu, Rui Zhang, Qunshu Lin, Bin Wang, Zhiyuan Zhao, Man Jiang, Xiaomeng Zhao, Jin Shi, Fan Wu, Pei Chu, Minghao Liu, Zhenxiang Li, Chao Xu, Bo Zhang, Botian Shi, Zhongying Tu, and Conghui He. OmniDocBench: Benchmarking diverse PDF document parsing with comprehensive annotations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Jake Poznanski, Luca Soldaini, and Kyle Lo. olmOCR 2: Unit test rewards for document OCR. *arXiv preprint arXiv:2510.19817*, 2025.

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

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Ray Smith. An overview of the Tesseract OCR engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007) Vol 2*, pp. 629–633, 2007. doi: 10.1109/ICDAR.2007.4376991. URL https://doi.org/10.1109/ICDAR.2007.4376991.

Bin Wang, Zhuangcheng Gu, Chao Xu, Bo Zhang, Botian Shi, and Conghui He. UniMER-Net: A universal network for real-world mathematical expression recognition. *arXiv preprint arXiv:2404.15254*, 2024a.

Bin Wang, Fan Wu, Linke Ouyang, Zhuangcheng Gu, Rui Zhang, Renqiu Xia, Bo Zhang, and Conghui He. CDM: A reliable metric for fair and accurate formula recognition evaluation. *arXiv preprint arXiv:2409.03643*, 2024b. URL https://arxiv.org/abs/2409.03643v1.

Bin Wang, Chao Xu, Xiaomeng Zhao, Linke Ouyang, Fan Wu, Zhiyuan Zhao, Rui Xu, Kaiwen Liu, Yuan Qu, Fukai Shang, et al. MinerU: An open-source solution for precise document content extraction. *arXiv preprint arXiv:2409.18839*, 2024c.

Bin Wang, Tianyao He, Linke Ouyang, Fan Wu, Zhiyuan Zhao, Tao Chu, Yuan Qu, Zhenjiang Jin, Weijun Zeng, Ziyang Miao, et al. MinerU2.5-Pro: Pushing the limits of data-centric document parsing at scale. *arXiv preprint arXiv:2604.04771*, 2026a.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. InternVL3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025. URL https://arxiv.org/abs/2508.18265.

Yinjie Wang, Ling Yang, Bowen Li, Ye Tian, Ke Shen, and Mengdi Wang. Revolutionizing reinforcement learning framework for diffusion large language models. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 101476–101501, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/a61023ce36d21010f1423304f8ec49af-Paper-Conference.pdf.

Haoran Wei, Yaofeng Sun, and Yukun Li. DeepSeek-OCR 2: Visual causal flow. *arXiv preprint arXiv:2601.20552*, 2026.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256, 1992. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Chengyue Wu, Hao Zhang, Shuchen Xue, Shizhe Diao, Yonggan Fu, Zhijian Liu, Pavlo Molchanov, Ping Luo, Song Han, and Enze Xie. Fast-dLLM v2: Efficient block-diffusion LLM. *arXiv preprint arXiv:2509.26328*, 2025.

Chengyue Wu, Shiyi Lan, Yonggan Fu, Sensen Gao, Jin Wang, Jincheng Yu, Jose M. Alvarez, Pavlo Molchanov, Ping Luo, Song Han, Ligeng Zhu, and Enze Xie. Fast-dVLM: Efficient block-diffusion VLM via direct conversion from autoregressive VLM. *arXiv preprint arXiv:2604.06832*, 2026.

Ling Yang, Ye Tian, Bowen Li, Xinchen Zhang, Ke Shen, Yunhai Tong, and Mengdi Wang. MMaDA: Multimodal large diffusion language models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2025a.

Zhibo Yang, Jun Tang, Zhaohai Li, Pengfei Wang, Jianqiang Wan, Humen Zhong, Xuejing Liu, Mingkun Yang, Peng Wang, Shuai Bai, Lianwen Jin, and Junyang Lin. CC-OCR: A comprehensive and challenging OCR benchmark for evaluating large multimodal models in literacy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 21744–21754, 2025b. doi: 10.1109/ICCV51701.2025.02019. URL https://openaccess.thecvf.com/content/ICCV2025/html/Yang_CC-OCR_A_Comprehensive_and_Challenging_OCR_Benchmark_for_Evaluating_Large_ICCV_2025_paper.html.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Xu Zhong, Elaheh ShafieiBavani, and Antonio Jimeno Yepes. Image-based table recognition: Data, model, and evaluation. In *European Conference on Computer Vision (ECCV)*, 2020. URL https://www.ecva.net/papers/eccv_2020/papers_ECCV/html/3802_ECCV_2020_paper.php.

Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei Chen, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. LLaDA 1.5: Variance-reduced preference optimization for large language diffusion models. *arXiv preprint arXiv:2505.19223*, 2025.

APPENDIX CONTENTS

A Theoretical Results **14**

A.1 Statistical Setting and Scope 14

A.2 Unbiased Joint Gradient with Leave-One-Out Rewards 15

A.3 Page-Level Objective Alignment 16

A.4 Noise in Layout Utility Estimates 16

A.5 Conditional Baselines and Finite-Sample Credit Assignment 16

A.6 Connection to HJPO and Ablation Results 18

B Related Work Extension **18**

B.1 Speculative Decoding for Document OCR 18

B.2 Direct Diffusion and Draft–Verify Trade-offs 19

B.3 Document Models and AR-to-Diffusion Conversion 19

B.4 Verifiable Document Rewards and Conditional Credit Assignment 20

C Additional Experimental Results **21**

C.1 Semantic Shuffle Analysis 21

D Prompt Design and Task Examples **22**

D.1 Layout Detection 22

D.2 Text Recognition 23

D.3 Formula Recognition 23

D.4 Table Recognition 24

E Experimental Details **25**

E.1 Training Overview 25

E.2 Autoregressive-to-Diffusion Continued Training (ADCT) 25

E.3 Reinforcement Learning Baselines and HJPO 26

E.4 Evaluation Details 29

E.5 HJPO Algorithm Pipeline 30

A THEORETICAL RESULTS

A.1 STATISTICAL SETTING AND SCOPE

We establish the statistical basis of page-level coupling and conditional credit assignment using score-function estimation (Williams, 1992) and baseline control variates (Greensmith et al., 2004). The analysis derives the joint gradient, quantifies noise in layout utility estimates, and characterizes the variance benefit of conditional baselines under nested layout–recognition sampling.

Fix a page x and a parameter value θ_0. Let $L \sim p_\theta(\cdot \mid x)$ and $O \sim q_\theta(\cdot \mid x, L)$. Here L and O denote complete sampled trajectories, including stochastic decisions; the reward depends on their decoded layout and recognition outputs. The formulation includes fully shared parameters for the

two distributions. Define

$$J_x(\theta) = \mathbb{E}_{p_\theta q_\theta}[Q(x, L, O)],$$
$$s_L = \nabla_\theta \log p_\theta(L \mid x)|_{\theta_0}, \qquad s_O = \nabla_\theta \log q_\theta(O \mid x, L)|_{\theta_0}. \tag{13}$$

All expectations below are at θ_0, conditional on x, unless stated otherwise. Assume parameter-independent support locally, differentiable probabilities or densities, and valid interchange of differentiation and integration. The reward Q has no explicit parameter dependence. Assume Q, the scores, and the reward–score products have the finite moments required below; in particular, all displayed variances and weighted second moments are finite. Rewards and sampled baselines are treated as constants when differentiating the score surrogate (stop-gradient).

Draw $K_L > 1$ independent layout groups. Within group j, draw $K_O > 1$ conditionally independent recognition trajectories, identically distributed under $q_{\theta_0}(\cdot \mid x, L_j)$. Different groups are independent. Write $Q_{jm} = Q(x, L_j, O_{jm})$, $\bar{Q}_j = K_O^{-1} \sum_m Q_{jm}$, and

$$B^L_{-j} = \frac{1}{K_L - 1} \sum_{k \neq j} \bar{Q}_k, \quad B^O_{j,-m} = \frac{1}{K_O - 1} \sum_{n \neq m} Q_{jn},$$
$$\mu(L) = \mathbb{E}[Q \mid x, L], \qquad \sigma^2(L) = \mathrm{Var}(Q \mid x, L). \tag{14}$$

The means and baselines match the unnormalized rewards in Section 3.2.

Theoretical flat comparator. The analytical flat comparator uses the same final reward Q for both scores and a deterministic page-only baseline $b_0(x)$, giving $(Q - b_0)(s_L + s_O)$. It targets the same J_x as the hierarchical estimator. The variance analysis considers its recognition contribution $(Q - b_0)s_O$. This deterministic-baseline comparison isolates the effect of conditioning the baseline on layout. The trained downstream-coupled GRPO comparator uses the sampled page-wide normalization defined in Appendix E.3.3.

A.2 Unbiased Joint Gradient with Leave-One-Out Rewards

Proposition A.1 (Joint score identity and hierarchical estimator)**.** *Under the assumptions above,*

$$\nabla J_x(\theta_0) = \mathbb{E}[Q(s_L + s_O)]. \tag{15}$$

The following estimator is unbiased for this shared-parameter gradient:

$$\widehat{g}_H = \frac{1}{K_L} \sum_j (\bar{Q}_j - B^L_{-j}) s_{L,j} + \frac{1}{K_L K_O} \sum_{j,m} (Q_{jm} - B^O_{j,-m}) s_{O,jm}. \tag{16}$$

Proof. Differentiating the joint density $p_\theta q_\theta$ gives $\nabla(p_\theta q_\theta) = p_\theta q_\theta (s_L + s_O)$ at θ_0. Integrating against Q proves Equation 15, including when the two distributions share parameters. Normalization of each distribution also gives

$$\mathbb{E}[s_L \mid x] = 0, \qquad \mathbb{E}[s_O \mid x, L] = 0. \tag{17}$$

Conditioning on L_j shows $\mathbb{E}[\bar{Q}_j s_{L,j}] = \mathbb{E}[\mu(L) s_L] = \mathbb{E}[Q s_L]$. The other layout groups are independent of L_j, hence $\mathbb{E}[B^L_{-j} s_{L,j}] = \mathbb{E}[B^L_{-j}]\mathbb{E}[s_{L,j}] = 0$. For recognition, $B^O_{j,-m}$ and O_{jm} are independent conditional on x, L_j, so

$$\mathbb{E}[B^O_{j,-m} s_{O,jm} \mid x, L_j] = \mathbb{E}[B^O_{j,-m} \mid x, L_j]\mathbb{E}[s_{O,jm} \mid x, L_j] = 0.$$

Each recognition summand therefore has expectation $\mathbb{E}[Q s_O]$. Averaging and adding the two terms proves the claim. Correlation between summands affects variance, not this expectation calculation. □

Subtracting a layout-dependent baseline preserves the recognition gradient because the recognition score has zero mean conditional on layout. Subtracting $\mu(L)$ from the layout reward removes its expected learning signal: $\mathbb{E}[(Q - \mu(L))s_L] = 0$. This conditional-expectation identity establishes the rationale for cross-layout comparisons in layout optimization and within-layout comparisons in recognition optimization.

A.3 PAGE-LEVEL OBJECTIVE ALIGNMENT

Local task rewards and final page-level parsing quality define distinct optimization objectives. The following smoothness result quantifies the local advantage of following the final objective's gradient over an equally sized direction with weaker alignment.

Proposition A.2 (Equal-norm local comparison)**.** *Let $g = \nabla J_x(\theta_0) \neq 0$. Suppose J_x has an H-Lipschitz gradient on a convex neighborhood containing the steps below. For any direction d with $\|d\| = \|g\|$, set $c = \langle g, d\rangle / \|g\|^2 \in [-1, 1]$. Then*

$$\begin{aligned} J_x(\theta_0 + \eta g) - J_x(\theta_0) &\geq \eta \|g\|^2 (1 - H\eta/2), \\ J_x(\theta_0 + \eta g) - J_x(\theta_0 + \eta d) &\geq \eta \|g\|^2 (1 - c - H\eta). \end{aligned} \tag{18}$$

Thus, for $H > 0$, $0 < \eta < 2/H$ guarantees improvement along g; if $c < 1$, $0 < \eta < (1 - c)/H$ guarantees a larger gain than along d, provided both steps stay in the neighborhood.

Proof. For any admissible displacement v, the fundamental theorem of calculus and Lipschitz continuity yield

$$|J_x(\theta_0 + v) - J_x(\theta_0) - \langle g, v\rangle| \leq \int_0^1 Ht\|v\|^2 \, dt = \frac{H}{2}\|v\|^2.$$

Use the lower bound with $v = \eta g$ for the first inequality. Subtract the upper bound for $v = \eta d$ from that lower bound and use equal norms for the second. Cauchy–Schwarz gives $c \leq 1$. If $H = 0$, the remainder vanishes and the corresponding conclusions hold directly without these divisions by H. □

The proposition applies to a nonzero direction from separate task rewards after rescaling it to norm $\|g\|$, with its alignment quantified by c. To illustrate objective mismatch, consider a hypothetical example with two layouts a, b, IoUs $1, 0.8$, and expected final utilities $\mu(a) = 0.2, \mu(b) = 0.9$. Let $p_\theta(a) = \rho = \mathrm{sigmoid}(\theta)$, keeping recognition fixed along this scalar parameter. Then

$$\frac{d}{d\theta}\mathbb{E}[\mathrm{IoU}] = 0.2\rho(1 - \rho) > 0, \qquad \frac{d}{d\theta} J_x = -0.7\rho(1 - \rho) < 0.$$

In this example, increasing geometric overlap decreases final parsing utility. Optimizing the common reward Q aligns both flat and hierarchical estimators with the final objective. Conditional credit assignment then determines how this shared reward is allocated across the two stages.

A.4 NOISE IN LAYOUT UTILITY ESTIMATES

Proposition A.3 (Nested reward variance)**.** *For a randomly sampled layout and its K_O recognition rollouts,*

$$\mathrm{Var}(\bar{Q} \mid x) = \mathrm{Var}_L(\mu(L) \mid x) + \frac{1}{K_O}\mathbb{E}_L[\sigma^2(L) \mid x]. \tag{19}$$

Proof. Conditional independence gives $\mathbb{E}[\bar{Q} \mid x, L] = \mu(L)$ and $\mathrm{Var}(\bar{Q} \mid x, L) = \sigma^2(L)/K_O$. Write $\bar{Q} - \mathbb{E}\bar{Q} = (\bar{Q} - \mu(L)) + (\mu(L) - \mathbb{E}\mu(L))$. The cross term has expectation zero after conditioning on L, since the first term has conditional mean zero. Taking squared expectations proves the identity. □

Averaging K_O recognition outcomes reduces the within-layout noise term by a factor of K_O, while preserving variation in expected utility across layouts. Equation 19 characterizes the scalar layout-utility estimate. In the ablation, downstream-coupled GRPO and HJPO both use $K_L = 4$ and $K_O = 8$, retaining the same layout and recognition rollout counts (Appendix E.3).

A.5 CONDITIONAL BASELINES AND FINITE-SAMPLE CREDIT ASSIGNMENT

For the recognition score $s = s_O$, define $w(L) = \mathbb{E}[\|s\|^2 \mid x, L]$ and assume $w(L) > 0$ almost surely. Layouts with $w = 0$ have a zero score almost surely and contribute zero to the gradient estimator. For a deterministic baseline $b(x, L)$ set $g_b = (Q - b)s$ and $V(g) = \mathrm{tr}\,\mathrm{Cov}(g \mid x)$.

All baselines below have finite weighted second moments. We minimize the score-weighted second moment that determines gradient variance, following the control-variate criterion (Greensmith et al., 2004).

Proposition A.4 (Oracle conditional baseline)**.** *The baseline*

$$b^*(L) = \frac{\mathbb{E}[Q\|s\|^2 \mid x, L]}{w(L)} \tag{20}$$

minimizes $V(g_b)$ over layout-conditioned deterministic baselines. For any such b, and in particular any page-only constant $b_0(x)$,

$$V(g_b) - V(g_{b^*}) = \mathbb{E}_L\big[w(L)(b(x,L) - b^*(L))^2 \mid x\big] \geq 0. \tag{21}$$

Proof. Equation 17 implies $\mathbb{E}[g_b \mid x, L] = \mathbb{E}[Qs \mid x, L]$, independently of b. Thus all baselines give the same mean, both conditionally and marginally. For fixed L, expand the conditional second moment:

$$\mathbb{E}[(Q-b)^2\|s\|^2 \mid x, L] = \mathbb{E}[Q^2\|s\|^2 \mid x, L] - 2b\, w(L)b^*(L) + b^2 w(L).$$

Completing the square gives its value at b^* plus $w(L)(b-b^*)^2$. Average over L and subtract the identical squared mean norm from both second moments to obtain the covariance-trace identity. □

The oracle baseline b^* weights rewards by the squared score norm. It reduces to the conditional reward mean $\mu(L)$ under the sufficient condition

$$\mathrm{Cov}(Q, \|s\|^2 \mid x, L) = 0 \quad \text{for almost every } L. \tag{22}$$

For example, a score norm constant across recognition outcomes at fixed x, L satisfies this condition. The following corollary establishes the finite-sample comparison under this additional assumption.

Corollary A.5 (Finite-sample benefit for one recognition contribution)**.** *Assume Equation 22. Let $\widehat{b}_{-m}$ be the average reward of the other $K_O - 1$ conditionally independent recognition rollouts at the same layout, and let $g_{\mathrm{LOO}} = (Q_m - \widehat{b}_{-m})s_m$. The best deterministic page-only baseline is*

$$b^*_{\mathrm{flat}}(x) = \frac{\mathbb{E}_L[w(L)\mu(L) \mid x]}{\mathbb{E}_L[w(L) \mid x]}. \tag{23}$$

Writing $g_{\mathrm{flat}} = (Q_m - b^*_{\mathrm{flat}})s_m$, we have

$$\begin{aligned} V(g_{\mathrm{flat}}) - V(g_{\mathrm{LOO}}) = \mathbb{E}_L[w(L)(\mu(L) - b^*_{\mathrm{flat}})^2 \mid x] \\ - \frac{\mathbb{E}_L[w(L)\sigma^2(L) \mid x]}{K_O - 1}. \end{aligned} \tag{24}$$

Consequently LOO has smaller covariance trace for this contribution precisely when the weighted between-layout term exceeds the baseline-estimation penalty.

Proof. Under the extra condition, $b^* = \mu$. Minimizing $\mathbb{E}_L[w(L)(\mu(L) - b_0)^2 \mid x]$ over b_0 by differentiation proves Equation 23. Conditional on x, L, the baseline error $\delta = \widehat{b}_{-m} - \mu(L)$ has mean zero, variance $\sigma^2(L)/(K_O - 1)$, and is independent of (Q_m, s_m). Expanding $g_{\mathrm{LOO}} = (Q_m - \mu)s_m - \delta s_m$, the cross term in its squared norm vanishes by this independence and zero mean. Hence

$$\mathbb{E}[\|g_{\mathrm{LOO}}\|^2 \mid x, L] = \mathbb{E}[\|(Q_m - \mu)s_m\|^2 \mid x, L] + \frac{w(L)\sigma^2(L)}{K_O - 1}.$$

The means are unchanged. Subtract this marginal variance from the oracle page-baseline variance in Equation 21 to obtain Equation 24. □

Equation 24 gives the exact variance difference for one recognition contribution under Equation 22. Conditional LOO reduces this variance precisely when weighted between-layout variation exceeds the finite-sample baseline-estimation cost. The comparison uses the optimal deterministic page-only baseline. For a minibatch average or a joint layout–recognition gradient, its variance additionally includes covariances between contributions that share samples.

A.6 Connection to HJPO and Ablation Results

Equation 16 establishes an unbiased joint estimator for full-trajectory scores, unnormalized LOO rewards, and on-policy sampling at the fixed parameter θ_0. The implemented HJPO updates optimize the regularized surrogate in Equation 8, using group-relative normalization (Shao et al., 2024), PPO clipping (Schulman et al., 2017), trajectory-length averaging, and a KL penalty. Layout and recognition updates are applied sequentially to the shared parameters. Appendix E.3 specifies the reward, sampling, normalization, and update definitions for the three trained configurations.

The analysis establishes complementary roles for the two components. The common reward Q aligns the ideal joint gradient with final parsing quality. Conditional baselines preserve that gradient and attain the variance benefit characterized by the preceding propositions and corollary under their stated assumptions.

Table 6 demonstrates the corresponding performance gains on OmniDocBench v1.5. Adding downstream coupling to Independent GRPO raises GT-layout Overall from 94.08 to 94.44 and Pred-layout Overall from 91.13 to 92.04. Adding conditional credit assignment further raises these scores to 94.82 and 92.63, respectively. Across the three configurations, the GT–Pred gap decreases from 2.95 to 2.40 to 2.19, and Reading Order edit distance decreases from 0.062 to 0.048 to 0.045. With the same Overall reward and $K_L = 4$, $K_O = 8$ rollout counts, HJPO outperforms downstream-coupled GRPO on both parsing settings and Reading Order. These ablations establish the empirical benefit of downstream coupling and conditional credit assignment in the evaluated configurations.

B Related Work Extension

This section situates our approach within document decoding research, focusing on candidate generation and verification, autoregressive-to-diffusion conversion, and learning from document-level rewards.

B.1 Speculative Decoding for Document OCR

Speculative decoding verifies multiple candidate tokens with an autoregressive (AR) target model, with exactness determined by the verification procedure (Leviathan et al., 2023). Under stochastic decoding, preserving the target distribution requires appropriate acceptance–rejection and residual-sampling rules. Under greedy decoding, accepting only the matching prefix preserves the fixed target model's greedy sequence. Multi-token prediction (MTP) supplies candidates; the verification rule determines whether these guarantees apply. Document OCR systems instantiate this framework through external pipelines, attached prediction heads, or diffusion-based candidate generators.

Document-pipeline drafting. HSD (Liao et al., 2026) uses PP-StructureV3 to generate layout regions and fixed drafts without additional training. It verifies regions in parallel before evaluating the assembled draft against the full page. Windowed alignment and prefix-tree evaluation handle discrepancies between the draft and target sequences. Its default rule accepts a draft token when its probability is at least 0.75 times that of the argmax token, allowing acceptance beyond exact argmax agreement. The reported near-lossless quality is therefore an empirical observation rather than an exact distributional or greedy-equivalence guarantee. With Transformers and FlexAttention on A100 GPUs, HSD reports a $2.42\times$ page-level speedup for dots.ocr on OmniDocBench v1.5, reaching $4.89\times$ on financial reports (Liao et al., 2026, Table 1).

Recursive-head drafting. Jina-OCR-v1 (Barón García et al., 2026) augments a DeepSeek-OCR-based decoder with a FastMTP block, recursively reused to predict three draft positions. The main decoder accepts the longest matching greedy prefix and supplies a correction or bonus token. Its lossless claim concerns greedy output; the procedure does not use stochastic residual resampling. On olmOCR-Bench with an NVIDIA L4, vLLM 0.20.1, and batch size one, the reported eager-mode speedup reaches $1.95\times$ at depth three. With CUDA graphs, the best reported speedup is $1.17\times$ at depth one, highlighting the dependence of realized gains on execution mode.

External diffusion drafting. HunyuanOCR-1.5 (Li et al., 2026) adapts DFlash with a separate 90.7M-parameter, five-layer block-diffusion drafter conditioned on target hidden states. The drafter

proposes a block, and the AR target verifies and accepts a valid prefix. The target model remains frozen throughout the training of the drafter. Under its single-request OmniDocBench protocol, the report gives speedups of $6.37\times$ with Transformers and $2.14\times$ with vLLM. These measurements characterize diffusion-assisted speculation within the respective systems. The cited speed table omits the accelerator model, limiting hardware-matched comparison with our measurements.

Shared-model diffusion drafting. GravityOCR (Kim et al., 2026) adapts GLM-OCR so that block-diffusion drafting and causal verification share parameters, requiring neither an independent drafter nor an auxiliary head. It retains an external layout pipeline and applies GRPO through the AR path. Using SGLang on one H100, it reports $3.94\times$ decode-only acceleration on region crops and $1.32\times$ full-page acceleration. The latter is measured on a 100-page subset with one page processed at a time. Across 8,922 English crops, the two serving paths produce exactly matching strings in 96.6% of cases. This measurement distinguishes mathematical greedy equivalence from numerical agreement between serving implementations. Together with external diffusion drafters, this design demonstrates how diffusion-based candidate generation can complement AR verification.

B.2 Direct Diffusion and Draft–Verify Trade-offs

Table 7 compares candidate generation and output acceptance across document decoders. In MinerU-Diffusion v2, bidirectional attention reconstructs masked tokens within a block, while completed blocks provide causal context. The diffusion decoding rule directly determines which predictions become output tokens, without a separate causal verification step. This supports element-specific block sizes and confidence thresholds within one checkpoint (Section 3.1). Direct diffusion and shared-model self-speculation both avoid maintaining an independent drafter–verifier pair, although they determine output acceptance differently.

Table 7: Candidate generation and output acceptance in document decoders. Entries summarize architectural mechanisms; efficiency depends on the evaluation conditions discussed in the text.

Method	Candidate generation	Output acceptance
HSD	External document pipeline	Region/page verification with threshold-based acceptance
Jina-OCR-v1	Recursive FastMTP head	Greedy prefix verification by the main decoder
HunyuanOCR-1.5	External diffusion drafter	AR target verification
GravityOCR	Shared block-diffusion path	Shared causal AR verification
MinerU-Diffusion v2	Block-diffusion decoder	Direct confidence-based denoising

Direct diffusion and draft–verify decoding expose different quality–latency trade-offs. Direct diffusion depends on denoising rounds, block sizes, and confidence thresholds, without guaranteeing preservation of the initializing AR distribution. Our ground-truth-layout results retain a quality gap to the stronger AR initializer. Verification introduces computation and may reject drafts, while preserving the specified target behavior when its exactness conditions hold. Bidirectional block prediction is available to both direct diffusion decoders and diffusion drafters. Our approach combines task-aligned conversion, adjustable direct block decoding, and downstream-coupled layout–recognition training.

B.3 Document Models and AR-to-Diffusion Conversion

Document-specialized AR models provide a starting point for adapting the decoding process while retaining task representations. MinerU2.5 separates layout analysis from recognition on high-resolution crops (Niu et al., 2025), while MinerU2.5-Pro emphasizes data-centric document training (Wang et al., 2026a). Our conversion retains this specialization, holding visual processing and task representations fixed while adapting the decoder and its training objectives.

Direct diffusion has also been investigated specifically for document OCR. DODO (Man et al., 2026) uses block discrete diffusion to address structural synchronization failures in global masked generation. MinerU-Diffusion (Dong et al., 2026) formulates OCR as inverse rendering with parallel

diffusion decoding. These methods establish direct diffusion as a document-parsing approach, with parallel generation operating without AR draft verification. Our work builds on this direction by converting an existing document-specialized AR parser and retaining adjustable inference block sizes.

AR-to-diffusion conversion has precedents in language and multimodal models. Dream (Ye et al., 2025) initializes language diffusion from pretrained AR weights. Fast-dLLM v2 (Wu et al., 2025) combines block diffusion, complementary attention, and caching to adapt AR language models. Fast-dVLM (Wu et al., 2026) extends conversion to pretrained multimodal models, including block-size annealing and both direct and self-speculative decoding. ADCT specializes this conversion framework to document parsing through the original task supervision and smaller-block replay. Subsequent HJPO training optimizes the coupled layout and recognition stages, connecting model conversion with downstream parsing quality.

B.4 Verifiable Document Rewards and Conditional Credit Assignment

Verifiable document rewards extend optimization beyond token-level supervision to properties of the generated output. olmOCR 2 (Poznanski et al., 2025) uses GRPO with binary unit tests for text, reading order, tables, and formulas. Jina-OCR-v1 uses dense verifiable rewards (Barón García et al., 2026), while GravityOCR optimizes OCR rewards through its causal verification path (Kim et al., 2026). These approaches illustrate complementary choices of reward structure and generation path for document optimization.

HJPO focuses on the dependency between layout selection and recognition conditioned on the selected layout. Layouts are compared through their expected downstream parsing utility, estimated from multiple recognition rollouts. Recognition trajectories are compared within each layout condition, yielding separate credit signals under a common page-level objective. This formulation combines document-level reward optimization with explicit conditioning on the layout that supplies recognition inputs.

C Additional Experimental Results

C.1 Semantic Shuffle Analysis

Vision-language OCR systems commonly pair a visual encoder with an autoregressive language decoder, making it difficult to distinguish faithful visual transcription from plausible completion driven by language priors. We therefore evaluate on Semantic Shuffle (Dong et al., 2026), which shuffles controlled proportions of words in 112 English pages from FOX (Liu et al., 2024) and re-renders the resulting text with consistent formatting. Across 13 perturbation levels, semantic coherence is progressively disrupted while the visual presentation remains comparable.

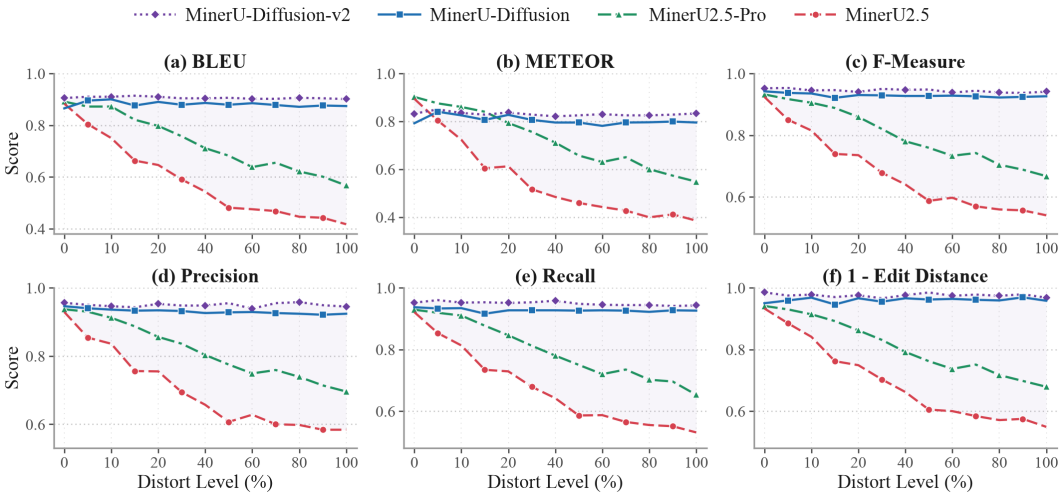


Figure 4: Semantic Shuffle results across perturbation levels. As word shuffling disrupts semantic coherence, the autoregressive parsers' scores decline sharply, whereas the two diffusion parsers remain comparatively stable. Shading illustrates the widening score gap between MinerU-Diffusion v2 and MinerU2.5.

Figure 4 shows contrasting responses across the evaluated decoding paradigms. Averaging the absolute change from 0% to 100% perturbation over the six metrics gives 0.009 for MinerU-Diffusion v2 and 0.012 for MinerU-Diffusion, compared with 0.287 for MinerU2.5-Pro and 0.413 for MinerU2.5. The latter two changes are approximately 32.5 and 46.8 times that of MinerU-Diffusion v2, respectively, using unrounded values. At 100% perturbation, MinerU-Diffusion v2 retains a BLEU score of 0.901 and a $1 - \text{edit distance}$ score of 0.968, exceeding MinerU2.5-Pro by 0.249–0.334 across all six metrics.

The autoregressive baseline is competitive when semantic coherence is preserved: at 0% perturbation, MinerU2.5-Pro scores higher than MinerU-Diffusion v2 on METEOR (0.902 versus 0.832), a metric that credits stem and synonym matches. From 20% perturbation onward, MinerU-Diffusion v2 instead achieves the higher METEOR score at every evaluated level. Among the four systems, MinerU-Diffusion v2 has the smallest mean absolute endpoint change, below that of the diffusion baseline MinerU-Diffusion. These results indicate low sensitivity to semantic perturbations on this benchmark.

D PROMPT DESIGN AND TASK EXAMPLES

This section provides the prompt formats, output specifications, and representative examples for the layout, text, formula, and table tasks of MinerU-Diffusion v2. These tasks share a unified prompt interface: a single `<image>` token followed by a plain-text task suffix, requiring no few-shot examples or structured metadata.

Task Prompt Convention. The four task suffixes and their output formats are summarized below:

- **Layout Detection** (Section D.1) localizes content regions and outputs bounding boxes with category labels and rotation flags.
- **Text Recognition** (Section D.2) transcribes cropped text regions into plain text.
- **Formula Recognition** (Section D.3) converts cropped formula regions into LaTeX markup.
- **Table Recognition** (Section D.4) serializes cropped tables into an OTSL-based token sequence with cell content, subsequently converted to HTML.

D.1 LAYOUT DETECTION

Layout Detection serves as the entry point of the document parsing pipeline, responsible for localizing all content regions on a page and assigning each a semantic category. The model takes a downsampled page image and produces a sequence of structured region descriptors.

Layout Detection: Prompt

```
<image>\nLayout Detection:
```

Output Format. The output is a newline-delimited sequence of region descriptors, where each region follows the format:

Layout Detection: Output Format

```
<|box_start|>x1 y1 x2 y2<|box_end|><|ref_start|>category<|ref_end|><|
  rotate_dir|>
```

Here, `x1 y1 x2 y2` are the normalized bounding box coordinates (scaled to $[0, 999]$), `category` is the semantic label of the region (e.g., title, text, header, footer, table, figure, formula), and `<|rotate_dir|>` indicates the text orientation (`<|rotate_up|>` for standard upright text, with other directions for rotated content). Regions are emitted in natural reading order (top-to-bottom, left-to-right for left-to-right scripts).

Example. Given a document title page, the model outputs:

Layout Detection: Output Example

```
<|box_start|>705 112 899 146<|box_end|><|ref_start|>header<|ref_end
  |><|rotate_up|>
<|box_start|>030 343 132 397<|box_end|><|ref_start|>title<|ref_end|><|
  rotate_up|>
<|box_start|>212 330 491 382<|box_end|><|ref_start|>title<|ref_end|><|
  rotate_up|>
<|box_start|>214 389 767 441<|box_end|><|ref_start|>title<|ref_end|><|
  rotate_up|>
<|box_start|>219 494 359 523<|box_end|><|ref_start|>text<|ref_end|><|
  rotate_up|>
<|box_start|>654 940 907 975<|box_end|><|ref_start|>footer<|ref_end
  |><|rotate_up|>
```

This output identifies six regions (one header, three title blocks, one text block, and one footer), each with precise spatial coordinates and upright orientation.

D.2 TEXT RECOGNITION

Text Recognition transcribes cropped text regions into plain text. Each region is an original-resolution crop produced by Stage 1 Layout Detection.

Text Recognition: Prompt

```
<image>\nText Recognition:
```

Output Format. The output is a plain-text string corresponding to the content of the cropped text region. No special tokens or markup are used. The model generates raw text as-is, including whitespace, punctuation, and any inline symbols present in the source image.

D.3 FORMULA RECOGNITION

Formula Recognition converts cropped formula regions into LaTeX markup. The model supports both inline and display-style formulas, as well as multi-line equation environments.

Formula Recognition: Prompt

```
<image>\nFormula Recognition:
```

Output Format. The output is a LaTeX math string. Display-style (block) formulas are wrapped in `\[...\]` delimiters. Equation numbers, when present in the source image, are preserved via `\tag{...}`. The model generates standard LaTeX math commands and environments (e.g., `\frac`, `\mathrm`, `\quad`), ensuring the output is directly compilable.

Example: Single Display Formula. Given a cropped equation region, the model outputs:

Formula Recognition: Output Example

```
\[L = 0.004 \ln (2D/d) \quad (\mu \mathrm{H} / \mathrm{cm}) \tag
  {4-2-10}\]
```

The equation number (4-2-10) from the original document is captured via `\tag{}`.

Multi-line Formulas. Multi-line equations are handled through the collaboration of Layout Detection and Formula Recognition. Layout Detection first identifies an equation block region encompassing the entire multi-line group, within which individual single-line formulas are separately localized. Each line is then independently cropped and recognized by Formula Recognition. The final multi-line output is produced by concatenating the individual LaTeX results in reading order, faithfully reproducing the original equation group without requiring the model to generate multi-line environments in a single pass.

D.4 TABLE RECOGNITION

Table Recognition converts cropped table regions into a structured token sequence based on OTSL (Optimized Table Structure Language) (Lysak et al., 2023). Cell content is transcribed as plain text, with inline formulas in LaTeX when present. The OTSL output is subsequently converted to HTML for downstream consumption.

Table Recognition: Prompt

```
<image>\nTable Recognition:
```

Output Format. The output is a flat token sequence representing the table structure row by row. Each cell is delimited by `<fcel>`, and rows are separated by `<nl>`. Cell content may contain plain text, LaTeX inline math (`\(...\)`), or a mixture of both. The OTSL representation is compact and unambiguous, supporting regular grids as well as cells with complex content. After generation, the OTSL sequence is programmatically converted to HTML for rendering and downstream integration.

Example. Given a cropped table with two rows (a header row of time values and a data row of concentration values), the model outputs:

Table Recognition: Output Example

```
<fcel>\( \frac{t}{\min} \)<fcel>3<fcel>5<fcel>7<fcel>10<fcel>15<fcel
  >21<fcel>25<nl>
<fcel>\( \frac{1/c}{{\mathrm{\;{mol}}}^{-1}\cdot {\mathrm{{dm}}}^{3}}
  \)<fcel>135.1<fcel>157.7<fcel>181.8<fcel>215.5<fcel>275.5<fcel
  >347.2<fcel>393.2<nl>
```

Each `<fcel>` token introduces a cell, and `<nl>` marks the end of a row. The first column contains LaTeX-formatted headers with units, while the remaining columns hold numeric values.

E EXPERIMENTAL DETAILS

E.1 TRAINING OVERVIEW

MinerU-Diffusion v2 is initialized from the 1.2B-parameter MinerU2.5-Pro checkpoint (Wang et al., 2026a), retaining its NaViT-675M vision encoder and Qwen2-0.5B language backbone. Training proceeds in two phases: ADCT converts the autoregressive parser using MinerU2.5-Pro's Stage-2 training data, followed by HJPO using its Stage-3 training data to jointly optimize layout generation and content recognition. Table 8 summarizes the configurations.

Table 8: Training setup and hyperparameters of MinerU-Diffusion v2 for ADCT and HJPO.

Category	**Parameter**	**ADCT**	**HJPO**
Vision	Max Resolution	$2048 \times 28 \times 28$	$2048 \times 28 \times 28$
	#Tokens per Image	64–2048	64–2048
Data	Dataset Type	Layout & OCR† & Image Analysis	Layout & Text & Formula & Table
	#Samples	3.9M (192K human-labeled)	192K
Model	Trainable	All	All
	Sequence Length	16384	8192
Training	Batch Size	256	64 pages
	ViT Learning Rate	5×10^{-6}	1×10^{-7}
	MLP/LLM Learning Rate	5×10^{-5}	1×10^{-5}
	Epoch	2	1

†OCR collectively refers to text recognition, formula recognition, and table recognition.

E.2 AUTOREGRESSIVE-TO-DIFFUSION CONTINUED TRAINING (ADCT)

ADCT preserves the initializer's visual architecture, tokenizer, task prompts, and output representations while adapting the decoder to block diffusion.

Training data. We reuse MinerU2.5-Pro's 3.9M deduplicated Stage-2 samples, covering layout, OCR, and image analysis, with a focus on Chinese and English. The training set comprises two parts: (1) 192K high-quality Hard samples produced through the expert annotation pipeline; and (2) replay data sampled proportionally from MinerU2.5-Pro's Stage-1 training set to prevent catastrophic forgetting. The Hard:Replay mixing ratios differ by subtask: $6{:}1$ for layout analysis, $1{:}50$ for text recognition, $1{:}25$ for formula recognition, $1{:}10$ for table recognition, and $1{:}4$ for image analysis. This non-uniform strategy reflects differences in hard-sample volume and baseline performance across subtasks. Layout analysis has more hard samples and a strong foundation from MinerU2.5-Pro's Stage-1 training, requiring less replay; text recognition has fewer hard samples and uses more replay to preserve generalization. The replay samples are already included in the Stage-2 mixture. The larger-recipe ablation adds MinerU2.5-Pro's Stage-1 data beyond this mixture. No new data are collected or synthesized.

Training configuration. All model parameters are trainable. We use a vision-encoder learning rate of 5×10^{-6} and an MLP/LLM learning rate of 5×10^{-5}, with a batch size of 256 and a maximum sequence length of 16,384 tokens. Training runs for two epochs over the 3.9M samples, giving 7.8M sample exposures. No additional VQA warm-up or post-conversion supervised fine-tuning is used.

Following Section 3.1, the primary block size grows from $L_B = 1$ to 64, then contracts to 16, without a separate stable-training or learning-rate annealing stage. During expansion, most samples use the primary size; a small fraction replay previously visited smaller sizes. Smaller-block replay continues at $L_B = 64$ and during subsequent training at $L_B = 16$; it is distinct from data replay. Each stage samples block sizes from its specified distribution and masks target tokens with probability $1 - \alpha_t$, minimizing the reconstruction objective in Equation 1.

E.3 Reinforcement Learning Baselines and HJPO

We compare three progressively extended reinforcement-learning configurations: Independent GRPO, GRPO with downstream coupling, and HJPO. They retain the underlying task-specific reward metrics and scoring rules, while changing reward aggregation, advantage comparison groups, and credit assignment. The configurations correspond to the GRPO, + Downstream coupling, and + Conditional credit assignment (HJPO) rows in Table 6, respectively.

E.3.1 Common Training Data and Configuration

Reinforcement learning starts from the ADCT checkpoint initialized from MinerU2.5-Pro and trained for two epochs on its Stage-2 data. Training reuses MinerU2.5-Pro's Stage-3 data, selected from a source pool of 192K expert-annotated samples.

We sample training trajectories using SGLang with a uniform block length $L_B = 8$ and a confidence threshold of 0.95. Sampling uses temperature 1.0, top-p $= 0.95$, and top-k $= 20$. All model parameters are trainable. The vision-encoder learning rate is 1×10^{-7}, and the MLP/LLM learning rate is 1×10^{-5}. We use a KL coefficient of $\beta = 0.01$, a batch size of 64 pages, a maximum sequence length of 8,192 tokens, and one training epoch.

The trajectory contraction factor is $s = 8$, and the average response length during reinforcement learning is 380. A length-based approximation gives $380/8 = 47.5 < 50$ forward computations per sample, or approximately 48.

E.3.2 Configuration 1: Independent GRPO

Training inputs and rewards. Independent GRPO (Shao et al., 2024) optimizes layout generation and content recognition using separate task-specific reward groups. Layout generation takes a full-page image as input and uses category IoU as its reward. Recognition uses ground-truth region crops, so its trajectories do not depend on sampled layout predictions. Text, formula, and table recognition use edit-distance-based rewards, CDM, and TEDS, respectively.

Within-input comparison. For each input, we sample $G = 32$ candidate outputs and compute group-relative advantages among candidates for that input. Layout candidates are compared within the same page; recognition candidates are compared within the same ground-truth crop and task. Layout rewards and advantages depend only on the layout metric and do not use downstream OCR scores. Following DAPO (Yu et al., 2025), the baseline uses clip-higher and dynamic sampling, discarding groups with zero reward variance.

E.3.3 Configuration 2: GRPO with Downstream Coupling

Extension from Configuration 1. Recognition inputs change from ground-truth crops to regions produced by sampled layouts, and both stages receive feedback from final page-level parsing quality. The layout signal consequently changes from category IoU to downstream Overall, while text, formula, and table scoring retain their original definitions. All parsing outcomes for a page form one comparison group for advantage estimation.

Common page-level reward. Both downstream-coupled GRPO and HJPO use the evaluation metric Overall as the downstream parsing reward, $Q(x, L, O) = \mathrm{Overall}(x, L, O)$. Here, x is the page image, L is the sampled layout, and O is the complete recognition output conditioned on that layout. We assemble the region outputs into a page prediction and compute

$$\begin{aligned} Q(x, L, O) &= \mathrm{Overall}(x, L, O) \\ &= \frac{(1 - \mathrm{TextEdit}) \times 100 + \mathrm{FormulaCDM} + \mathrm{TableTEDS}}{3}, \end{aligned} \tag{25}$$

where the component scores are evaluated on (x, L, O) under the rules in Appendix E.4. FormulaCDM and TableTEDS use percentage scores, and Q retains the original Overall scale without an additional division by 100. Layout IoU, Reading Order, and TEDS-S are not separate terms in this reward. Independent GRPO retains category IoU for its layout task; the coupled configurations

instead provide downstream feedback through Overall. Matching, component aggregation, and handling of invalid or missing outputs follow the evaluation rules. Configuration 3 retains this reward definition.

Hierarchical sampling. For each page, the frozen old policy samples $K_L = 4$ layout trajectories. For each layout L_j, it generates $K_O = 8$ OCR trajectories from the predicted regions, yielding 32 complete page-parsing outcomes:

$$M_{m,j} = Q(x, L_j, O_{m,j}), \qquad j \in \{1, \dots, 4\}, \quad m \in \{1, \dots, 8\}. \tag{26}$$

Each $O_{m,j}$ represents a complete page recognition outcome under L_j, rather than the output for a single region.

Page-level advantage normalization. We compute the reward mean and standard deviation over all 32 outcomes for the same page:

$$\mu_x = \frac{1}{32} \sum_{j=1}^{4} \sum_{m=1}^{8} M_{m,j}, \qquad \sigma_x = \sqrt{\frac{1}{32} \sum_{j=1}^{4} \sum_{m=1}^{8} (M_{m,j} - \mu_x)^2}. \tag{27}$$

Each OCR trajectory receives the advantage

$$\widetilde{A}^{O}_{m,j} = \frac{M_{m,j} - \mu_x}{\sigma_x + \delta}, \tag{28}$$

where $\delta > 0$ is a numerical stabilizer. Thus OCR trajectories are compared across all sampled layouts of the same page.

Downstream feedback for layout generation. We average the page rewards from the eight OCR outcomes for each layout:

$$\bar{M}_j = \frac{1}{8} \sum_{m=1}^{8} M_{m,j}. \tag{29}$$

The layout advantage uses the same page-level mean and standard deviation:

$$\widetilde{A}^{L}_{j} = \frac{\bar{M}_j - \mu_x}{\sigma_x + \delta} = \frac{1}{8} \sum_{m=1}^{8} \widetilde{A}^{O}_{m,j}. \tag{30}$$

A layout therefore receives the average feedback from its eight recognition outcomes. Layout and OCR share the reward source and normalization scale, but generally receive different numerical advantages. We do not further normalize the four layout advantages or separately center and scale OCR rewards within each layout.

Policy updates. The layout rollout set $\widetilde{\mathcal{T}}_L$ contains four trajectories per page, weighted by $\widetilde{A}^{L}_{j}$. The OCR rollout set $\widetilde{\mathcal{T}}_O$ contains 32 trajectories per page, weighted by $\widetilde{A}^{O}_{m,j}$. Using the contracted-trajectory objective in Section 3.2, we update the shared parameters as

$$\begin{aligned} \theta' &= \theta + \eta_L \nabla_\theta \mathcal{J}^{s_L}(\theta; P_L, \widetilde{\mathcal{T}}_L), \\ \theta^{+} &= \theta' + \eta_O \nabla_{\theta'} \mathcal{J}^{s_O}(\theta'; P_O, \widetilde{\mathcal{T}}_O). \end{aligned} \tag{31}$$

The layout update precedes the OCR update, which uses the updated shared parameters. The old policy, sampled trajectories, histories, and advantages remain fixed during both updates. The old policy is refreshed before the next rollout iteration. This configuration uses the same trajectory contraction, slice-wise computation, and clipped objective as HJPO, with globally normalized advantages replacing the hierarchical advantages.

E.3.4 CONFIGURATION 3: HJPO WITH CONDITIONAL CREDIT ASSIGNMENT

Extension from Configuration 2. HJPO retains the predicted-layout inputs, Overall reward, and hierarchical rollout counts of Configuration 2. It replaces page-wide comparison with separate comparisons across layouts and among OCR trajectories within each layout. Each page uses $K_L = 4$ layouts and $K_O = 8$ OCR trajectories per layout, sampled by the same frozen old policy.

Conditional reward construction. For each layout, the mean downstream reward estimates its expected parsing utility:

$$\bar{M}_j = \frac{1}{K_O} \sum_{m=1}^{K_O} M_{m,j}. \tag{32}$$

HJPO constructs separate leave-one-out rewards for layout and recognition:

$$\begin{aligned} R_j^L &= \bar{M}_j - \frac{1}{K_L - 1} \sum_{j' \neq j} \bar{M}_{j'}, \\ R_{m,j}^O &= M_{m,j} - \frac{1}{K_O - 1} \sum_{m' \neq m} M_{m',j}. \end{aligned} \tag{33}$$

Group-specific normalization. Layout rewards are normalized across layout candidates for the same page. OCR rewards are normalized separately within each layout condition. Layout updates therefore compare downstream utility across layouts, whereas OCR updates compare recognition outcomes under a fixed layout.

Retained update procedure. HJPO retains the shared-policy alternating updates, trajectory contraction, and slice-wise computation of Configuration 2, replacing the globally normalized advantages with conditional advantages. Layout is updated first, followed by OCR, while the old policy remains fixed during both updates. Appendix E.5 provides the complete procedure.

E.3.5 PROGRESSION ACROSS THE THREE CONFIGURATIONS

Moving from Configuration 1 to Configuration 2 replaces independent task rewards with a common page-level reward and exposes OCR to predicted-layout inputs. Layout updates can consequently use downstream recognition quality. Moving from Configuration 2 to Configuration 3 retains the reward and rollout counts: four layouts and eight OCR trajectories per layout yield 32 page-parsing outcomes. HJPO restricts OCR comparisons to each layout and normalizes layout and OCR credit separately. Table 9 summarizes the progression.

Table 9: The three reinforcement-learning configurations in Table 6. Configurations 2 and 3 share the Overall reward and hierarchical rollout counts.

Aspect	1: Independent GRPO	2: Downstream-coupled GRPO	3: HJPO
OCR inputs	Ground-truth crops	Regions from sampled layouts	Regions from sampled layouts
Sampling	32 candidates per input	4 layouts/page; 8 OCR trajectories/layout	4 layouts/page; 8 OCR trajectories/layout
Layout feedback	Category IoU	Mean downstream Overall; page-level normalization	Mean downstream Overall; cross-layout LOO comparison and normalization
OCR comparison group	32 candidates for the same ground-truth crop	All 32 page-parsing outcomes	8 OCR outcomes under the same layout
Increment	Independent task optimization	Common page reward and predicted-layout inputs	Conditional grouping and stage-specific credit

E.4 Evaluation Details

Benchmarks and task settings. For full-document parsing and layout analysis, we evaluate our models on OmniDocBench v1.5 (Ouyang et al., 2025). Table recognition is assessed using CC-OCR (Yang et al., 2025b) and OCRBench v2 (Fu et al., 2025), while formula recognition is evaluated on UniMER-Test (Wang et al., 2024a). The inference prompts for these tasks are summarized in Appendix D. Unless otherwise stated, all OmniDocBench results use the inference settings described below and follow the OmniDocBench v1.5 evaluation protocol on 1,355 pages with hybrid matching. On OmniDocBench, text is evaluated by edit distance (↓), formulas by CDM (↑), and tables by TEDS / TEDS-S (↑). The Overall score is computed from the three core parsing metrics:

$$\text{Overall} = \frac{(1 - \text{TextEdit}) \times 100 + \text{FormulaCDM} + \text{TableTEDS}}{3}.$$

Reading Order and TableTEDS-S are auxiliary metrics and are not included in Overall. Reading Order is only reported under the w/o GT Layout setting, where the model must jointly predict layout and content from the full page. Under w/ GT Layout, oracle layout regions are provided, so the evaluation mainly isolates recognition quality after removing layout detection errors.

Decoding configurations. All evaluation experiments use top-$k = 0$, temperature 1.0, and top-$p = 1.0$. We adapt SGLang as the inference engine for MinerU-Diffusion v2 to accelerate inference and evaluation. The MinerU-Diffusion v2 results in Section 4.2 use element-aware block lengths $4/8/8$ and confidence thresholds $0.92/0.96/0.92$, both in formula/table/text order.

The conversion ablation in Table 4 evaluates uniform inference block lengths $L' \in \{8, 16, 32, 64\}$ with a confidence threshold of 0.95. The ground-truth-layout comparison in Table 5 uses uniform block lengths $8/8/8$, $16/16/16$, and $32/32/32$ at the same threshold. Element-aware configurations use block lengths $4/8/8$, $4/8/16$, and $4/16/8$, with thresholds $0.92/0.96/0.92$ in formula/table/text order. Within this comparison, the checkpoint is fixed while inference block allocation and thresholds are varied, as described in Section 4.3.2.

Inference efficiency. All evaluations conducted by us use NVIDIA H200 GPUs. Table 3 reports tokens committed per forward pass (TPF) and effective output tokens per second (TPS), alongside the corresponding OmniDocBench v1.5 score. TPF measures decoding parallelism, whereas TPS reports effective output throughput. The starred olmOCR 2 and DeepSeek-OCR 2 entries use official decoding results where ground-truth-layout decoding could not be run, as marked in the table.

Semantic perturbation evaluation. Semantic Shuffle uses 112 English FOX pages at 13 word-shuffle levels: 0%, 5%, 10%, 15%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, and 100%. We report BLEU, METEOR, F-Measure, Precision, Recall, and $1 -$ edit distance at each level. The endpoint summary is the equal-weight mean of the six absolute score changes between 0% and 100%. Benchmark construction, individual metric curves, and the associated analysis appear in Appendix C.1.

E.5 HJPO ALGORITHM PIPELINE

Algorithm 1 | **HJPO** (Hierarchical Joint Policy Optimization)

Input: Page dataset $\mathcal{D}$; page reward $Q(x, L, O)$; shared policy π_θ.
Prompts P_L, P_O; rollout group sizes $K_L, K_O > 1$; ADCT checkpoint θ_{ADCT}.
Contraction factors s_L, s_O; PPO clip ϵ; KL weight β; step sizes η_L, η_O.
Initialize: $\theta \leftarrow \theta_{\mathrm{ADCT}}$.
for each page minibatch $\mathcal{B} \subset \mathcal{D}$ during training **do**
  Freeze a copy $\theta_{\mathrm{old}} \leftarrow \theta$; set $\mathcal{T}_L \leftarrow \varnothing$, $\mathcal{T}_O \leftarrow \varnothing$.
  Sample hierarchical trajectories with the frozen old policy:
  for each page $x \in \mathcal{B}$ **do**
    for $j = 1, \ldots, K_L$ **do**
      Sample $\tau_j^L \sim \pi_{\theta_{\mathrm{old}}}(\cdot \mid x, P_L)$; decode layout L_j.
      for $m = 1, \ldots, K_O$ **do**
        Sample $\tau_{m,j}^O \sim \pi_{\theta_{\mathrm{old}}}(\cdot \mid x, L_j, P_O)$; decode content $O_{m,j}$.
        $M_{m,j} \leftarrow Q(x, L_j, O_{m,j})$.
      end for
    end for
    Construct conditional rewards and advantages for this page:
    $\bar{M}_j \leftarrow K_O^{-1} \sum_{m=1}^{K_O} M_{m,j}$ for each j.
    $R_j^L \leftarrow \bar{M}_j - (K_L - 1)^{-1} \sum_{j' \neq j} \bar{M}_{j'}$ for each j.
    $R_{m,j}^O \leftarrow M_{m,j} - (K_O - 1)^{-1} \sum_{m' \neq m} M_{m',j}$ for each (m, j).
    Normalize $\{R_j^L\}_{j=1}^{K_L}$ across layouts to obtain $\{A_j^L\}_{j=1}^{K_L}$.
    For each j, normalize $\{R_{m,j}^O\}_{m=1}^{K_O}$ to obtain $\{A_{m,j}^O\}_{m=1}^{K_O}$.
    Append $\{(x, \tau_j^L, A_j^L)\}_j$ to $\mathcal{T}_L$.
    Append $\{((x, L_j), \tau_{m,j}^O, A_{m,j}^O)\}_{m,j}$ to $\mathcal{T}_O$.
  end for
  Keep θ_{old}, sampled trajectories, histories, and advantages fixed below.
  Contract trajectories and update the shared policy in order:
  for $p = L$ then $p = O$ **do**
    Group every s_p consecutive diffusion steps *within each block* into a segment.
    Preserve action order and the final shorter segment (Eq. 5).
    Set $S_{i,r}^p = \lceil B_{i,r}^p / s_p \rceil$; denote segment ℓ by $z_{i,r,\ell}^p$ (Eq. 11).
    for $\ell = 1, \ldots, S_{\max}^p$, where $S_{\max}^p = \max_{i,r} S_{i,r}^p$ **do**
      Batch valid segments $\{z_{i,r,\ell}^p : \ell \leq S_{i,r}^p\}$ at this slice position.
      Evaluate segment probabilities in parallel under block-causal attention.
      Use recorded histories and within-segment factorization for both policies.
      Form current/old ratios (Eq. 6) and current-to-old KL terms.
    end for
    Sum $C_\epsilon(r, A) - \beta D_{\mathrm{KL}}$ over valid segments, retaining the trajectory-length and group normalization of $\mathcal{J}^{s_p}(\theta; P_p, \mathcal{T}_p)$ (Eq. 8).
    $\theta \leftarrow \theta + \eta_p \nabla_\theta \mathcal{J}^{s_p}(\theta; P_p, \mathcal{T}_p)$.
  end for
  Refresh the old policy only at the next rollout iteration.
end for
Output: Shared policy π_θ for layout generation and content recognition.