MUTUALLY ADVERSARIAL SELF-TRAINING WITH EVOLVING DATA FOR UNIFIED MULTIMODAL MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Unified multimodal models (UMMs) combine image generation and visual understanding in a shared backbone. Since generation and understanding are inverse tasks, recent studies self-train UMMs by letting the two branches *cooperatively* supervise each other. We introduce MATE (*Mutually Adversarial self-Training with Evolving data*), a reinforcement-learning-based post-training framework in which the two branches instead *challenge* each other, and the challenges evolve as the model trains. MATE lets generation and understanding take turns to be challenger and solver. Given an image, the understanding branch proposes several candidate descriptions that the generation branch must turn back into similar images, and vice versa. The candidates are screened for consistency with the image or prompt they were proposed from, and the solver is trained on the candidate it handles worst. The adversary thus comes from the model's own outputs, and no separate adversary is trained. Moreover, the candidates that defeat one branch become the sources of the next challenges to the other in the next epoch, which keeps the challenges evolving with the model and turns the training into self-play in data space. On Janus-Pro-1B, MATE improves GenEval by 2.4 points, DPG-Bench by 1.7 points, and the average over nine understanding benchmarks by 0.7 points, while strengthening consistency across repeated image-text cycles.

1 INTRODUCTION

Unified multimodal models (UMMs) integrate visual understanding and generation into a single backbone (Zhao et al., 2026; Xie et al., 2024), promising mutually reinforcing capabilities and a single interface for both. UMM training still follows the pretrain-then-finetune recipe, so UMM capabilities are bounded by the coverage of externally annotated data. Self-training (Deng et al., 2025b; Yang et al., 2026), on the other hand, is a natural way past this bound, and the unified architecture opens a new route to it. Understanding and generation are inverse tasks handled by two branches of one model, so each branch can supervise the other without new labels. Recent work builds on this duality by letting one branch score, filter, or supply training data for the other (Hong et al., 2025; Qu et al., 2025; Jin et al., 2026; Yang et al., 2025; Pan et al., 2026; Mao et al., 2025; Fang et al., 2026). In most of such work, the two branches *cooperate*, agreeing on what counts as good and helping each other do more of it on whatever examples happen to be sampled.

In this paper, we argue that the two branches can also *challenge* each other. First, adversarial pressure directs learning toward cases on which the model performs poorly, whereas a cooperative loop tends to reinforce what the model already does well. Second, a UMM is the natural place for such a contest. Its two capabilities are inverses of each other, so each branch is a ready-made adversary for the other. Moreover, because they share one backbone, what one learns from its hard cases is, at least in principle, available to the other.

Our key idea is therefore to *pit the two branches of a UMM against each other, with generation and understanding taking turns as challenger and solver*. The generator's images challenge the understanding branch to describe them, and the understanding branch's descriptions challenge the generator to realize them. Some adversarial frameworks introduce a separately trained component, such as a discriminator or perturber (Saxena & Cao, 2021; Su et al., 2026). Here, neither branch is trained to challenge. Each produces its ordinary outputs, and the challenge is whichever of them the other branch handles worst, so the adversary lives in *data* and consists only of these selected outputs.

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

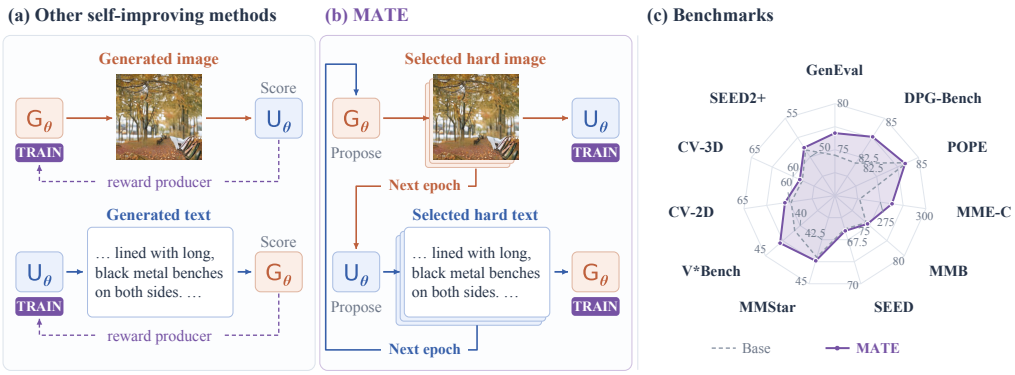


Figure 1: **Self-improvement through mutual challenges.** (a) In other self-improving methods such as SUDER (Hong et al., 2025), one branch scores an output to reward its producer. (b) Our MATE selects valid outputs that challenge the opposite branch, trains that solver, and uses the selected outputs to construct new challenges in the next epoch. In both panels, G_θ and U_θ share one backbone and the examples are real training records. (c) MATE and Janus-Pro-1B on the 11 benchmarks of the main tables. Each axis uses its own truncated linear range, shared by both methods, so polygon areas do not measure overall performance. MME-C is in cognition points, and other scores are percentages.

The next question is how an adversary made of data can supply a continuing training signal, since a selected challenge is consumed by one update and then spent. We take from self-play (Zhao et al., 2025; Huang et al., 2026) the principle that the opponent improves as the player does. Unlike prior work, which keeps the opponent in a second network (Goodfellow et al., 2014) or an archive of past checkpoints (Vinyals et al., 2019; OpenAI et al., 2019), we propose to keep it in an evolving input pool of the model's own selected outputs. The image that defeated the understanding branch becomes the source of the next challenges to the generator, and vice versa. This evolution keeps the challenge moving with the model, whereas a fixed pool pins it near where it started and runs out of pressure once its sources are mastered. We call this *self-play in data space*.

Our approach: We propose *Mutually Adversarial self-Training with Evolving data* (MATE), a post-training framework that combines mutual challenge between the two branches of a UMM with self-play in data space (Figure 1). MATE keeps the objective and the optimizer of standard reinforcement-learning post-training and changes only where the inputs of each branch come from and what its outputs are scored against. Each epoch plays one round at every source in the input pools. A round starts from a source text or a source image, and the challenger proposes several candidates in the other modality. Only the candidates that agree with the source are admitted, so a candidate the solver fails on is hard rather than invalid. The solver attempts each admitted candidate several times, and each attempt is scored on its match to the candidate and its consistency with the source. The candidate with the lowest mean score is the challenge, and the solver is updated with group-relative policy optimization (GRPO) (Shao et al., 2024) on the attempts that measured it. The selected challenges of one epoch are the sources of the next, so an output that defeated one branch becomes the source of new challenges to the other. Our contributions are:

- **Adversarial self-training for UMMs.** To our knowledge, MATE is the first post-training framework in which the two branches of a UMM challenge each other, with both branches trained and no adversary beyond the model itself.
- **Self-play in data space.** We propose a self-play scheme that evolves the model's opponent in data, as a pool of its own hardest admitted outputs, rebuilt every epoch from the outputs that defeated one branch to challenge the other.
- **Empirical study.** On Janus-Pro-1B (Chen et al., 2025), MATE improves generation and average understanding over the base model and outperforms two self-training baselines on both. Ablations indicate that challenge selection and evolving pools contribute to the generation gains.

2 Related Work

We relate MATE to four lines of prior work, and Table 3 in Appendix A compares it with the closest methods along the axes discussed below.

Unified multimodal models. UMMs place visual understanding and image generation in one backbone (Zhao et al., 2026; Xie et al., 2024; Chen et al., 2024a; Wang et al., 2026b). They differ in how they represent and generate images. Fully autoregressive models predict discrete visual tokens in sequence (Chameleon Team, 2024; Wang et al., 2024b; Chen et al., 2025). Show-o (Xie et al., 2025b) combines autoregression with discrete diffusion over those tokens inside a single transformer. Other designs add a flow-matching head to the backbone (Xie et al., 2025c), connect a pretrained multimodal model to a diffusion model (Wu et al., 2025b), or assign understanding and generation to separate transformer experts (Deng et al., 2025a). MATE uses a backbone that both produces images and describes them; our experiments use the fully autoregressive Janus-Pro.

Cooperative self-training for UMMs. Self-training without new annotation is studied for language models (Yang et al., 2026; Tao et al., 2024) and for multimodal models (Deng et al., 2025b). In a UMM the generation and understanding branches are inverses, so each can supply supervision for the other. SUDER (Hong et al., 2025) and GvU (Pan et al., 2026) score an output by the likelihood that the reverse task assigns to the original input. SUDER applies this in both directions and trains both branches, whereas GvU scores only generated images and trains only the generator. UniRL (Mao et al., 2025), SILMM (Qu et al., 2025), and HermesFlow (Yang et al., 2025) check a generated image against questions whose answers follow from the prompt. UniRL constructs the prompts and trains both branches on the correctness of the answers, SILMM lets the model write both the prompts and the questions, and HermesFlow trains the generator to prefer the image with the most correct answers over the image with the fewest. SRUM (Jin et al., 2026) and UniCorn (Fang et al., 2026) have the understanding branch rate the generated images. SRUM rates an image as a whole and region by region, and UniCorn also lets the model write the prompts and fine-tunes on its best-rated images together with captioning, judgment, and reflection data. These methods use cross-modal agreement to score outputs or build preference pairs. MATE selects source-consistent candidates that the opposite branch handles poorly, then carries these challenges into the next epoch's source pools.

Adversarial training. Adversarial training exposes a model's weaknesses with challenges that are optimized against it (Goodfellow et al., 2014; 2015; Madry et al., 2018; Razaviyayn et al., 2020) or selected from a pool because the model already finds them hard (Shrivastava et al., 2016). On understanding tasks, recent multimodal work obtains such challenges from a challenger model that mines hard negatives for a solver (Qiu et al., 2026) or from a multi-player game (Wang et al., 2026a). For UMMs, UniGame (Su et al., 2026) attaches a perturber at the shared token interface, constrains its perturbations around the embeddings of real images, retains hard examples in a buffer, and trains the understanding branch on labeled question-answer pairs. MATE constructs challenges in both directions and optimizes both tasks, whereas UniGame optimizes the understanding objective and a learned perturber. MATE draws its challenges from candidates the branches already produce and needs no annotation beyond the source itself, so it introduces no component beyond the model.

Self-play and evolving training data. Self-play trains a model against opponents derived from itself (Tesauro, 1995; Silver et al., 2017; Chen et al., 2024c), and methods differ in where the opponent is kept. It can be a population of current and past agents (Vinyals et al., 2019; OpenAI et al., 2019), a proposer trained to pose problems for the solver (Zhao et al., 2025; Huang et al., 2026), or a buffer of training cases that is replayed and expanded (Jiang et al., 2021b;a; Parker-Holder et al., 2022). EVA (Ye et al., 2025) is the closest of the buffer methods to MATE, since it evolves its most informative prompts into new ones. Self-play is already claimed for UMM post-training (Yang et al., 2025; Fang et al., 2026; Su et al., 2026), where the opponent is a role, a module, or an earlier round of the same model. MATE instead keeps the opponent in its input pools, where a description or an image that one branch handles worst becomes a source of challenges for the other. EVA returns its evolved prompts to the same solver in the same modality, whereas in MATE both alternate from one epoch to the next.

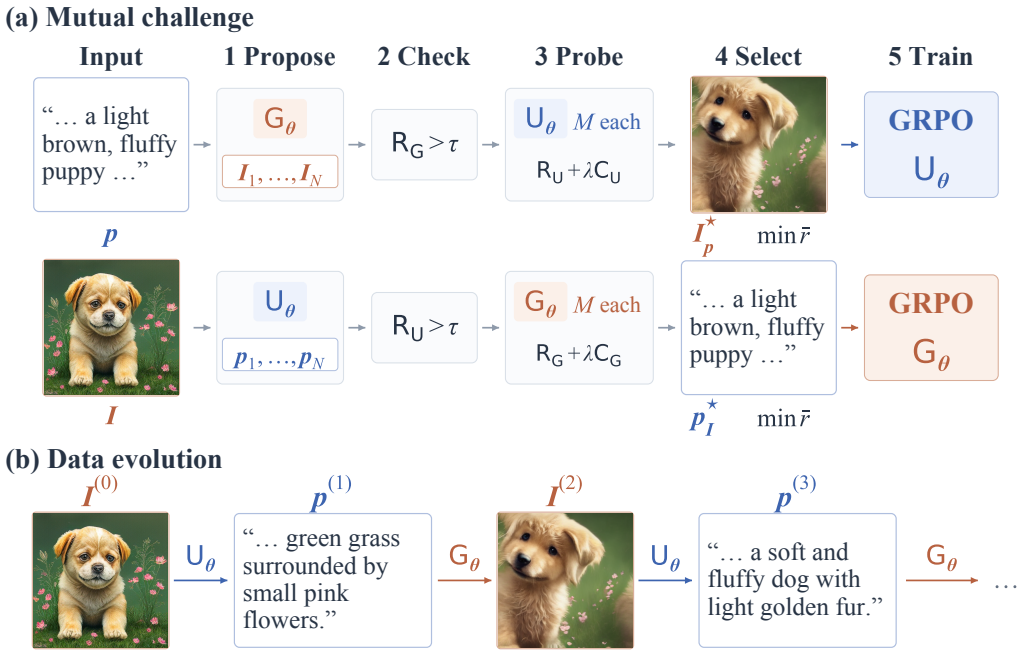


Figure 2: **MATE: mutual challenge and data evolution. (a) Mutual challenge.** The lower path exchanges the challenger and solver roles of the upper path. (1) The challenger proposes N candidates. (2) A fixed scoring function admits candidates above τ, or the highest-scoring candidate if none passes. (3) The solver makes M attempts per admitted candidate, scored by its scoring function R and input-consistency term $\lambda\mathsf{C}$. (4) The lowest mean score $\bar{r}$ identifies the selected challenge. (5) GRPO reuses all M existing attempts for that challenge. Inputs and selected challenges are records from (b), and other candidates and attempts are shown symbolically. **(b) Data evolution.** Consecutive image and text records from one training trajectory illustrate $\boldsymbol{I}^{(0)} \to \boldsymbol{p}^{(1)} \to \boldsymbol{I}^{(2)} \to \boldsymbol{p}^{(3)}$. Each arrow applies the procedure in (a) and advances one epoch, so a selected output becomes the input of the opposite branch. Quoted text is taken from actual model outputs and shortened for display.

3 Method

MATE trains the generation and understanding branches of a UMM to improve through challenges produced by each other. We first define the standard reinforcement learning objective (Section 3.1), then describe how to select challenging inputs and update the branch that solves them (Section 3.2). Finally, we explain how selected outputs form the next epoch's input pools, allowing the model to generate new challenges as training progresses (Section 3.3).

3.1 Preliminaries

A UMM has generation and understanding branches that share a backbone. The generator $\mathsf{G}_{\boldsymbol{\theta}}$ produces an image from a text prompt, and the understanding branch $\mathsf{U}_{\boldsymbol{\theta}}$ produces a description of an image. Throughout, $\boldsymbol{p}$ denotes a text prompt or description and $\boldsymbol{I}$ denotes an image. In this paper, we consider post-training both branches with reinforcement learning, which updates a branch on scored samples of its own outputs. Outputs are scored by two fixed scoring functions, one per branch. The scoring function $\mathsf{R}_\mathsf{G}(\boldsymbol{I}, \boldsymbol{p})$ measures the agreement of an image with a text, and $\mathsf{R}_\mathsf{U}(\boldsymbol{p}, \boldsymbol{I})$ that of a text with an image. In both, the sampled output is written first and the reference second.

For a branch $\pi \in \{\mathsf{G}, \mathsf{U}\}$ with an input $\boldsymbol{x}$ and a reward function r on its outputs, the objective is the expected reward $\mathsf{J}(\boldsymbol{\theta}; \pi, \boldsymbol{x}, \mathsf{r}) = \mathbb{E}_{\boldsymbol{y} \sim \pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x})}[\mathsf{r}(\boldsymbol{y})]$, which is maximized over $\boldsymbol{\theta}$ using sampled outputs. The standard procedure is direct post-training on a corpus $\mathcal{D} = \{(\boldsymbol{p}_i, \boldsymbol{I}_i)\}$. For every pair it maximizes $\mathsf{J}(\boldsymbol{\theta}; \mathsf{G}, \boldsymbol{p}_i, \mathsf{R}_\mathsf{G}(\cdot, \boldsymbol{p}_i))$ for the generator and $\mathsf{J}(\boldsymbol{\theta}; \mathsf{U}, \boldsymbol{I}_i, \mathsf{R}_\mathsf{U}(\cdot, \boldsymbol{I}_i))$ for the understanding branch, so each output is scored against the input it was produced from. The corpus fixes the inputs in

advance, regardless of how well either branch already performs on them. MATE instead trains each branch on inputs that challenge it. It keeps the objective J and the optimizer of the standard procedure and differs from it only in where the inputs come from and what the outputs are scored against.

3.2 MUTUAL CHALLENGE

Direct post-training draws the inputs of a branch from the corpus. MATE instead seeks an input that challenges the branch under training. Since the two branches of a UMM are inverse tasks and the outputs of one are inputs of the other, the key idea of MATE is to treat those outputs as candidates and to train a branch on the candidate that yields its lowest score. One branch serves as the challenger that proposes candidates and the other as the solver that is trained on them (Figure 2(a)). The roles can be exchanged, so the challenge is mutual.

Formulation. Take the understanding branch $\mathsf{U}_{\boldsymbol{\theta}}$ as the challenger and the generator $\mathsf{G}_{\boldsymbol{\theta}}$ as the solver. Given a source image $\boldsymbol{I}$, the challenge is the candidate on which the solver has the lowest expected score,

$$\boldsymbol{p}^{\star}_{\boldsymbol{I}} \;=\; \operatorname*{arg\,min}_{\boldsymbol{p}\in\mathcal{V}_{\boldsymbol{\theta}}(\boldsymbol{I})} \; \mathsf{s}_{\boldsymbol{\theta}}(\boldsymbol{p};\boldsymbol{I}), \tag{1}$$

where $\mathsf{s}_{\boldsymbol{\theta}}(\boldsymbol{p};\boldsymbol{I})$ is the solver's expected reward on a candidate, and $\mathcal{V}_{\boldsymbol{\theta}}(\boldsymbol{I})$ is the set of candidates the challenger may pose, consisting of its own descriptions that satisfy a consistency threshold with the source image,

$$\mathcal{V}_{\boldsymbol{\theta}}(\boldsymbol{I}) \;=\; \big\{\, \boldsymbol{p} \in \operatorname{supp}\, \mathsf{U}_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{I}) \;:\; \mathsf{R}_{\mathsf{U}}(\boldsymbol{p},\boldsymbol{I}) > \tau \,\big\}. \tag{2}$$

Candidates are outputs of the current challenger, and challenge selection operates on these outputs without training a separate adversary. We treat a candidate as valid if its agreement with the source image exceeds a threshold τ. Applying this validity criterion before evaluating difficulty reduces the risk of selecting a candidate merely because it misrepresents the source image.

The solver's score is the expected reward of its attempts at the candidate, where each attempt is also rewarded for consistency with the source image,

$$\mathsf{s}_{\boldsymbol{\theta}}(\boldsymbol{p};\boldsymbol{I}) \;=\; \mathbb{E}_{\hat{\boldsymbol{I}}\sim\mathsf{G}_{\boldsymbol{\theta}}(\cdot\mid\boldsymbol{p})}\big[\, \mathsf{R}_{\mathsf{G}}(\hat{\boldsymbol{I}},\boldsymbol{p}) + \lambda\, \mathsf{C}_{\mathsf{G}}(\hat{\boldsymbol{I}},\boldsymbol{I}) \,\big], \tag{3}$$

where $\hat{\boldsymbol{I}}$ is the solver's attempt at $\boldsymbol{p}$, C_{G} is a frozen consistency measure between two images, and λ is its weight. The first term measures whether the generated image matches the candidate description, and the consistency term encourages it to preserve the content of the source image. This additional reference is available from the challenge construction and requires no new annotation.

When the understanding branch is the solver, the challenge starts from a source text $\boldsymbol{p}$. The generator's candidates are its images $\boldsymbol{I}$ with $\mathsf{R}_{\mathsf{G}}(\boldsymbol{I},\boldsymbol{p}) > \tau$, and the understanding branch's score on a candidate is the expected value of $\mathsf{R}_{\mathsf{U}}(\hat{\boldsymbol{p}},\boldsymbol{I}) + \lambda\, \mathsf{C}_{\mathsf{U}}(\hat{\boldsymbol{p}},\boldsymbol{p})$ over its descriptions $\hat{\boldsymbol{p}} \sim \mathsf{U}_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{I})$, where C_{U} is the consistency measure between two texts.

Implementation. In practice, MATE does not solve Equation 1 directly, because enumerating all possible challenger outputs is intractable. We approximate the search by sampling candidates and solver outputs. With the generator as the solver, a round starting from a source image $\boldsymbol{I}$ samples N candidates $\boldsymbol{p}_1,\ldots,\boldsymbol{p}_N \sim \mathsf{U}_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{I})$ and admits $\mathcal{F} = \{\boldsymbol{p}_n : \mathsf{R}_{\mathsf{U}}(\boldsymbol{p}_n,\boldsymbol{I}) > \tau\}$. If no candidate satisfies the threshold, $\mathcal{F}$ is set to contain only the candidate with the highest $\mathsf{R}_{\mathsf{U}}(\boldsymbol{p}_n,\boldsymbol{I})$. For each admitted candidate $\boldsymbol{p}$ it draws M rollouts $\hat{\boldsymbol{I}}_m \sim \mathsf{G}_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{p})$ with rewards $r_m = \mathsf{R}_{\mathsf{G}}(\hat{\boldsymbol{I}}_m,\boldsymbol{p}) + \lambda\, \mathsf{C}_{\mathsf{G}}(\hat{\boldsymbol{I}}_m,\boldsymbol{I})$, and their mean $\bar{r}(\boldsymbol{p}) = \frac{1}{M}\sum_{m=1}^{M} r_m$ estimates the candidate's score of Equation 3. The round selects $\boldsymbol{p}^{\star}_{\boldsymbol{I}} = \operatorname{arg\,min}_{\boldsymbol{p}\in\mathcal{F}} \bar{r}(\boldsymbol{p})$ as the challenge, approximating the search in Equation 1 with sampled candidates and scores. The solver is then updated to maximize the objective $\mathsf{J}(\boldsymbol{\theta};\mathsf{G},\boldsymbol{p}^{\star}_{\boldsymbol{I}},\mathsf{r}^{\star})$ with the reward function $\mathsf{r}^{\star} = \mathsf{R}_{\mathsf{G}}(\cdot,\boldsymbol{p}^{\star}_{\boldsymbol{I}}) + \lambda\, \mathsf{C}_{\mathsf{G}}(\cdot,\boldsymbol{I})$, so any policy-gradient method applies. We use GRPO (Shao et al., 2024), which treats the rollouts of an input as a group and standardizes their rewards within it. The M rollouts already drawn for $\boldsymbol{p}^{\star}_{\boldsymbol{I}}$ serve as its group, so the same samples estimate the candidate's difficulty and provide the solver's training update. Appendix B gives the details of the update and the full procedure.

3.3 SELF-PLAY IN DATA SPACE

Section 3.2 defines a single round at a fixed source image or text. A fixed source bounds the difficulty that can be proposed from it, so rounds over a fixed corpus supply no further pressure once the model handles everything the corpus can pose. MATE therefore lets the inputs evolve with the model.

In MATE, self-play means that the model generates training challenges for its own two branches and uses the selected outputs to construct future challenges. Each branch runs one challenge-selection round for every input in its pool (Figure 2(b)). Let $\mathcal{P}_k^p$ and $\mathcal{P}_k^I$ denote the pools of input texts and images at epoch k, initialized as $\mathcal{P}_0^p = \{\boldsymbol{p}_i\}$ and $\mathcal{P}_0^I = \{\boldsymbol{I}_i\}$ from the corpus. Each round selects a description $\boldsymbol{p}_{\boldsymbol{I}}^\star$ from a source image $\boldsymbol{I}$, or an image $\boldsymbol{I}_{\boldsymbol{p}}^\star$ from a source text $\boldsymbol{p}$. These selected candidates form the input pools for the next epoch,

$$\mathcal{P}_{k+1}^I \;=\; \left\{\, \boldsymbol{I}_{\boldsymbol{p}}^\star \;:\; \boldsymbol{p} \in \mathcal{P}_k^p \,\right\}, \qquad \mathcal{P}_{k+1}^p \;=\; \left\{\, \boldsymbol{p}_{\boldsymbol{I}}^\star \;:\; \boldsymbol{I} \in \mathcal{P}_k^I \,\right\}, \tag{4}$$

so an output selected for its difficulty to one branch becomes the source of new challenges for the other. Repeating this update produces a sequence that alternates between modalities, $\boldsymbol{p}^{(0)} \to \boldsymbol{I}^{(1)} \to \boldsymbol{p}^{(2)} \to \cdots$, with one transition per epoch. Each round uses only a source text or image, so the procedure does not require paired data. Updating the pools allows the range of possible challenges to change with the model, whereas fixed inputs restrict the search to candidates derived from the original corpus.

We refer to this procedure as *self-play in data space*: the evolving input pools and the current model's outputs supply the training challenges. Selected outputs carry information about each branch's current difficulties into the next epoch, allowing the challenge distribution to adapt without a separate opponent network or an archive of past checkpoints.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Model & data. We instantiate MATE on Janus-Pro-1B (Chen et al., 2025), an autoregressive UMM with a 1.5B-parameter language backbone. We use 5000 image-text pairs from the text-to-image split of Align-Anything (Ji et al., 2024) to initialize text and image pools. Training does not use the original pairings, and none of the corpus texts appears among the prompts of the generation benchmarks. Both scoring functions are implemented with the frozen reward model UnifiedReward (Wang et al., 2026c), which takes no part in evaluation. The consistency measure is the similarity between SigLIP features. Appendix C specifies the prompts and normalization of both scores.

Training. A round samples $N = 4$ candidates and draws $M = 8$ rollouts per admitted candidate, with threshold $\tau = 0.5$, consistency weight $\lambda = 0.1$, and standard-deviation floor $\epsilon = 10^{-6}$. Both branches are updated by GRPO on all 2.1B model parameters with learning rate 10^{-6}, clipping range 0.2, and reference penalty 0.01. Training runs for five epochs with an effective batch size of 48 on six GPUs. MATE takes about 29 hours because each input produces N candidates and up to NM rollouts. All training methods and ablations share the optimizer, scoring functions, corpus, and batch size. Each reported result comes from one training run, so we interpret sub-point GenEval differences cautiously.

Baseline methods. We compare against two groups of baselines. The first consists of the self-training methods UniRL (Mao et al., 2025) and SUDER (Hong et al., 2025), which we re-implement on Janus-Pro-1B using their published training procedures. The second consists of published generation, understanding, and unified models. We report their generation scores from their papers without prompt rewriting. For understanding, we use scores from the OpenVLM leaderboard, which uses the same evaluation toolkit, when available, and evaluate the remaining model–benchmark combinations ourselves. We evaluate Janus-Pro-1B and all three methods trained from it under the same protocol; bold marks the best result among these four.

Benchmarks. Generation is measured on GenEval (Ghosh et al., 2023) and DPG-Bench (Hu et al., 2024). We use the official GenEval evaluation code and the ELLA evaluation code for DPG-Bench, generating four images per prompt for each benchmark. GenEval reports the mean of its

Table 1: Main results on image generation.

Method	# Params	GenEval↑ (per category)						**GenEval↑**	DPG-Bench↑
		Single Obj.	Two Obj.	Counting	Colors	Position	Color Attr.	**Overall**	Overall
Generation only									
SD3-Medium (Esser et al., 2024)	2B	99.0	94.0	72.0	89.0	33.0	60.0	74.0	84.1
LlamaGen (Sun et al., 2024)	0.8B	71.0	34.0	21.0	58.0	7.0	4.0	32.0	–
Sana-0.6B (Xie et al., 2025a)	0.6B	99.0	76.0	64.0	88.0	18.0	39.0	64.0	83.6
Sana-1.6B (Xie et al., 2025a)	1.6B	99.0	77.0	62.0	88.0	21.0	47.0	66.0	84.8
Lumina-Image 2.0 (Qin et al., 2025)	2.6B	–	87.0	67.0	–	–	62.0	73.0	87.2
UMMs									
Janus (Wu et al., 2025a)	1.3B	97.0	68.0	30.0	84.0	46.0	42.0	61.0	79.7
JanusFlow (Ma et al., 2025)	1.3B	97.0	59.0	45.0	83.0	53.0	42.0	63.0	80.1
Show-o (Xie et al., 2025b)	1.3B	98.0	80.0	66.0	84.0	31.0	50.0	68.0	67.3
Janus-Pro-1B (Chen et al., 2025)	1.5B	98.4	84.1	53.1	91.2	63.8	56.0	74.4	82.1
+ UniRL (Mao et al., 2025)	1.5B	97.5	82.8	51.6	**92.0**	**66.5**	60.5	75.2	82.1
+ SUDER (Hong et al., 2025)	1.5B	**99.4**	83.6	53.1	90.4	63.2	56.0	74.3	82.0
+ MATE (ours)	1.5B	99.1	**87.9**	**53.8**	90.7	65.5	**64.0**	**76.8**	**83.8**

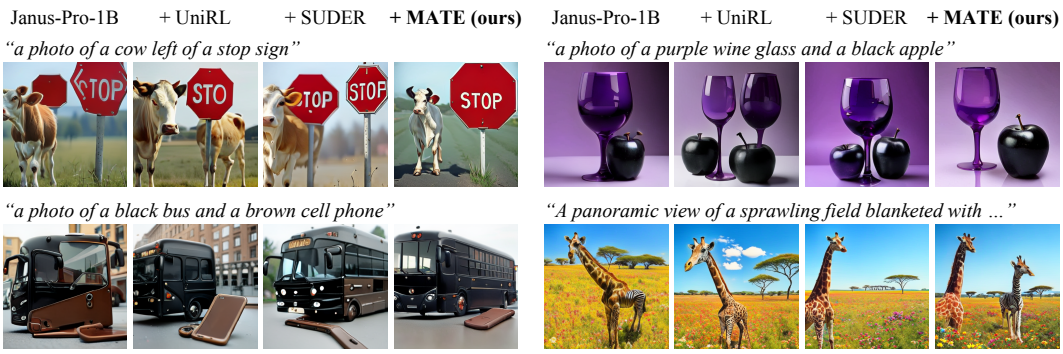


Figure 3: **Generation samples.** Images for the same prompt from Janus-Pro-1B and the three self-training methods applied to it; three prompts are from GenEval and one from DPG-Bench. One selected sample per model and prompt is shown from four generated samples, using the same sampling settings. The DPG-Bench prompt asks for a giraffe and a zebra standing side by side in a field of wildflowers; its displayed text is shortened.

six category accuracies. For Janus-Pro-1B and its post-trained variants, images are sampled at 384 pixels with classifier-free guidance weight 5 and temperature 1. Understanding is measured with VLMEvalKit (Duan et al., 2024) using benchmark-specific scoring; Janus-based evaluations use the Janus-Pro chat template and greedy decoding. We report overall F1 on POPE (Li et al., 2023b), the cognition score of MME (Fu et al., 2025), MMBench (Liu et al., 2025), the image split of SEED-Bench (Li et al., 2023a), MMStar (Chen et al., 2024b), V*Bench (Wu & Xie, 2024), the 2D and 3D splits of CV-Bench (Tong et al., 2024), and SEED-Bench-2-Plus (Li et al., 2024). MMBench reports accuracy on the English development questions, which we evaluate for every model. The first five columns cover general understanding, and the last four assess fine-grained visual perception relevant to image description. The understanding average is the mean over the nine columns after the MME cognition score is rescaled to $[0, 100]$ by its maximum of 800.

4.2 Generation and Understanding

Image generation. Table 1 reports generation results. MATE raises the GenEval score of Janus-Pro-1B by 2.4 points and its DPG-Bench score by 1.7, outperforming both self-training baselines on both benchmarks. The largest GenEval category gains are in two-object generation and color attribution, which require distinguishing objects and their attributes. Figure 3 compares selected outputs for the same prompts across all four methods.

Visual understanding. Table 2 reports understanding results. MATE raises the average from 56.9 to 57.6 and improves eight of the nine benchmarks relative to the base model, with the largest gain on MME cognition; POPE decreases by 0.1 points. UniRL and SUDER fall below the base model on most understanding benchmarks. Together, these results show improvements in both image generation and average visual understanding under MATE. Figure 4 shows examples.

Table 2: Main results on visual understanding. Higher is better on every benchmark.

Method	# Params	POPE	MME-C	MMB	SEED	MMStar	V*Bench	CV-2D	CV-3D	SEED2+	**Avg.**
Understanding only											
LLaVA-v1.5 (Liu et al., 2024)	7B	86.1	302.1	72.9	65.8	33.1	45.0	62.0	64.3	41.3	56.5
DeepSeek-VL-1.3B (Lu et al., 2024)	1.3B	85.9	225.0	74.5	66.0	39.9	42.9	66.2	62.7	43.7	56.7
LLaVA-OV-0.5B (Li et al., 2025)	0.5B	87.8	237.5	67.1	66.6	37.7	39.3	38.6	50.6	52.6	52.2
UMMs											
Janus (Wu et al., 2025a)	1.3B	84.1	238.6	70.0	62.9	37.6	41.4	49.1	58.7	38.5	52.5
JanusFlow (Ma et al., 2025)	1.3B	85.4	268.2	76.9	70.3	47.2	44.5	61.9	67.8	45.9	59.3
Show-o (Xie et al., 2025b)	1.3B	82.4	238.2	50.9	54.6	29.6	37.7	34.2	49.1	23.3	43.5
Janus-Pro-1B (Chen et al., 2025)	1.5B	**84.3**	263.6	74.5	66.9	43.5	40.8	59.8	58.9	50.9	56.9
+ UniRL (Mao et al., 2025)	1.5B	83.8	258.9	73.4	65.7	42.0	42.4	57.0	55.1	49.2	55.7
+ SUDER (Hong et al., 2025)	1.5B	83.5	255.4	73.5	65.8	**44.5**	40.3	59.6	56.8	50.5	56.3
+ MATE (ours)	1.5B	84.2	**281.4**	**74.7**	**67.0**	43.7	**42.9**	**60.5**	**59.2**	**51.2**	**57.6**



Q: Is there a cup in the image?
Janus-Pro-1B: Yes ✗
+ UniRL: Yes ✗
+ SUDER: Yes ✗
+ MATE (ours): No ✓

Q: Is there a bicycle in the image?
Janus-Pro-1B: No ✗
+ UniRL: No ✗
+ SUDER: No ✗
+ MATE (ours): Yes ✓

Q: What is the man on the sidelines holding?
Janus-Pro-1B:
A can of beer ✗
+ UniRL: A can of beer ✗
+ SUDER:
A can of beer ✗
+ MATE (ours):
A water bottle ✓

Q: Is the blue luggage on the left or right side of the bus?
Janus-Pro-1B: left ✗
+ UniRL: left ✗
+ SUDER: left ✗
+ MATE (ours): right ✓

Q: Which is closer to the camera, the bookcase (red box) or the books (blue box)?
Janus-Pro-1B:
bookcase ✗
+ UniRL: bookcase ✗
+ SUDER: bookcase ✗
+ MATE (ours):
books ✓

Q: Which is closer to the car (red box), the truck (blue box) or the trailer (green box)?
Janus-Pro-1B: trailer ✗
+ UniRL: trailer ✗
+ SUDER: trailer ✗
+ MATE (ours): truck ✓

Figure 4: **Understanding samples.** Six questions from POPE, SEED-Bench, V*Bench, and CV-Bench. Multiple-choice answers show the selected option's text; three questions are shortened and images cropped for display. A check mark denotes agreement with the reference answer.

4.3 Ablation and Analysis

Challenge selection and evolving inputs. Figure 6(b) compares MATE with three variants. It reports the average over the 9 understanding benchmarks (Table 2) and the average of GenEval and DPG-Bench. Keeping the initial pools fixed lowers the generation average of MATE from 80.3 to 79.9, and additionally selecting the highest-scoring candidate instead of the hardest (cooperative selection) lowers it to 79.5, compared with 78.3 for the base model. The understanding averages of MATE and these two variants range from 57.6 to 57.8, all above the base model's 56.9. Thus, challenge selection and evolving inputs account for part of the generation gain, whereas the understanding gain does not depend on them. Each difference is 0.4 points and comes from a single run, so these results are suggestive rather than conclusive. Removing the admission threshold from MATE, so that candidates are not checked against τ as in Equation 2, lowers both averages, to 57.3 and 79.3. Table 5 in Appendix C evaluates three more variants, which remove the consistency term, train on the groups of all admitted candidates, or use separate networks for the two branches.

Evolving training examples. Figure 5 follows one example through five epochs, and Figure 7 in Appendix C shows a second. The selected images and descriptions change as the input pools are updated, illustrating how MATE constructs subsequent inputs from its own outputs.

Round-trip consistency. Starting from 32 held-out Align-Anything test images, we apply k rounds of description and regeneration. Figure 6(a) reports two mean cosine similarities, each computed with one fixed encoder for all methods. On the image side, we compare the original image with the

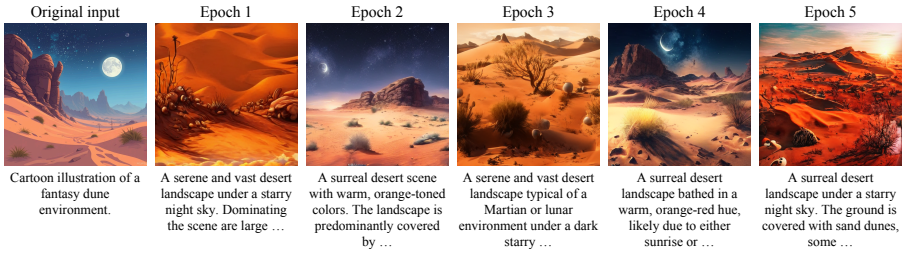
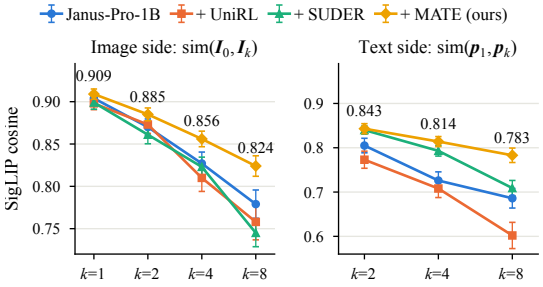


Figure 5: **An evolving training input under MATE.** An original dataset pair (left) is followed through five epochs. Selected images and descriptions become inputs for the next epoch. The image and text in each column come from separate training directions and need not form a matched pair. Descriptions are shortened for display. Figure 7 in Appendix C shows a second example.

(a) Round-trip consistency



(b) Ablation of MATE

	Und. Avg.↑	Gen. Avg.↑
Janus-Pro-1B	56.9	78.3
MATE (ours)	57.6	80.3
MATE w/o evolving pools	57.7	79.9
MATE w/o adversarial training	57.8	79.5
MATE w/o admission threshold	57.3	79.3

Figure 6: **Round-trip consistency and ablation.** (a) Left: SigLIP cosine between the original image $\boldsymbol{I}_0$ and the image $\boldsymbol{I}_k$ after k rounds. Right: SigLIP text cosine between the first description $\boldsymbol{p}_1$ and the k-th description $\boldsymbol{p}_k$. Points are means over 32 held-out samples and whiskers are standard errors. (b) Understanding and generation averages of Janus-Pro-1B, MATE, and three variants of MATE.

image after k rounds, using the SigLIP (Zhai et al., 2023) vision encoder of the base UMM. On the text side, we compare the first description with the k-th description, using the text encoder of the same SigLIP model family. Appendix C gives further details. On the image side, MATE exceeds the base model at every evaluated number of rounds. After eight rounds, its similarity is 0.824, compared with 0.779 for the base model, 0.758 for UniRL, and 0.745 for SUDER. On the text side, MATE also exceeds all three at every evaluated number of rounds, with 0.783 after eight rounds compared with 0.686 for the base model, 0.602 for UniRL, and 0.709 for SUDER. These results indicate that MATE better preserves content over repeated image–text cycles on both sides. Because MATE's consistency term is also a SigLIP similarity, this evaluation is not fully independent of its training reward.

5 Conclusion

We present MATE, a post-training framework that uses the generation and understanding branches of a UMM to challenge each other. By checking candidates for source consistency and selecting those the opposite branch handles worst, MATE focuses training on the model's current weaknesses without a separately trained adversary or new annotations. Selected images and descriptions become the next epoch's inputs, allowing challenges to evolve with the model. On Janus-Pro-1B, MATE improves both image generation and average visual understanding over the base model and two self-training baselines, while better preserving content through repeated image–text cycles. Ablations indicate that hard challenge selection and evolving inputs contribute to the generation gains.

Limitations. Due to computational constraints, our experiments use a 1B-scale UMM and 5000 training pairs, and we plan to evaluate MATE with larger UMMs and larger, higher-quality training datasets in future work. MATE also approximates the search in Equation 1 with a few sampled candidates and rollouts, so the selected challenge may not be the optimal one. Proposing and scoring these candidates also adds computation relative to other post-training methods, which more efficient candidate screening could reduce.

AI USE STATEMENT

In this work, we used generative AI tools (LLM-based coding assistants) to implement and debug parts of the training and evaluation code, to write scripts for launching, monitoring, and aggregating experiments, to draft data analysis scripts, and to edit the manuscript for readability. We have not used generative AI tools to formulate the research hypotheses, to design the method, or to generate synthetic training data beyond the self-play procedure described in Section 3; assisting in the writing of proofs is not applicable to this work. We have reviewed all AI-assisted work: LLM-generated code was inspected and tested by the authors, all reported numbers were recomputed from raw evaluation outputs under the protocol in Section 4, and all text was checked by the authors. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

All experiments build on publicly available models and datasets. The training recipe, hyperparameters, and the evaluation protocol (official GenEval and DPG-Bench evaluation code, VLMEvalKit with the Janus-Pro chat template and greedy decoding) are described in Section 4 and the appendix; training scripts, configuration files, and evaluation code will be released as supplementary material, together with the recorded candidate scores and selections of the self-play loop used for the qualitative analysis.

REFERENCES

Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024.

Liang Chen, Zekun Wang, Shuhuai Ren, Lei Li, Haozhe Zhao, Yunshui Li, Zefan Cai, Hongcheng Guo, Lei Zhang, Yizhe Xiong, Yichi Zhang, Ruoyu Wu, Qingxiu Dong, Ge Zhang, Jian Yang, Lingwei Meng, Shujie Hu, Yulong Chen, Junyang Lin, Shuai Bai, Andreas Vlachos, Xu Tan, Minjia Zhang, Wen Xiao, Aaron Yee, Tianyu Liu, and Baobao Chang. Next token prediction towards multimodal intelligence: A comprehensive survey. arXiv preprint arXiv:2412.18619, 2024a.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are we on the right way for evaluating large vision-language models? In *Advances in Neural Information Processing Systems*, volume 37, pp. 27056–27087. Curran Associates, Inc., 2024b.

Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-Pro: Unified multimodal understanding and generation with data and model scaling. arXiv preprint arXiv:2501.17811, 2025.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 6621–6642. PMLR, 21–27 Jul 2024c.

João Coelho, João Magalhães, Bruno Martins, and Chenyan Xiong. Effective reinforcement learning for agentic search by recycling zero-variance queries during training. arXiv preprint arXiv:2606.10709, 2026.

Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, Guang Shi, and Haoqi Fan. Emerging properties in unified multimodal pretraining. arXiv preprint arXiv:2505.14683, 2025a.

Shijian Deng, Kai Wang, Tianyu Yang, Harsh Singh, and Yapeng Tian. Self-improvement in multimodal large language models: A survey. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 1987–2006, Suzhou, China, November 2025b. Association for Computational Linguistics.

Haodong Duan, Junming Yang, Yuxuan Qiao, Xinyu Fang, Lin Chen, Yuan Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, Dahua Lin, and Kai Chen. VLMEvalKit: An open-source toolkit for evaluating large multi-modality models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, MM '24, pp. 11198–11201. ACM, October 2024.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 12606–12633. PMLR, 21–27 Jul 2024.

Zhen Fang, Ruiyan Han, XinYu Sun, Yuchen Ma, Ziheng Wang, Yu Zeng, Zehui Chen, Lin Chen, Wenxuan Huang, Wei-Jie Xu, Yi Cao, and Feng Zhao. UniCorn: Towards self-improving unified multimodal models through self-generated supervision. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13638–13669, San Diego, California, United States, July 2026. Association for Computational Linguistics.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, Rongrong Ji, Caifeng Shan, and Ran He. MME: A comprehensive evaluation benchmark for multimodal large language models. In *Advances in Neural Information Processing Systems*, volume 38. Curran Associates, Inc., 2025.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. GenEval: An object-focused framework for evaluating text-to-image alignment. In *Advances in Neural Information Processing Systems*, volume 36, pp. 52132–52152. Curran Associates, Inc., 2023.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, pp. 2672–2680. Curran Associates, Inc., 2014.

Jixiang Hong, Yiran Zhang, Guanzhong Wang, Yi Liu, Ji-Rong Wen, and Rui Yan. SUDER: Self-improving unified large multimodal models for understanding and generation with dual self-rewards. arXiv preprint arXiv:2506.07963, 2025.

Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. ELLA: Equip diffusion models with LLM for enhanced semantic alignment. arXiv preprint arXiv:2403.05135, 2024.

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. R-Zero: Self-evolving reasoning LLM from zero data. In *International Conference on Learning Representations*, pp. 130770–130790, 2026.

Jiaming Ji, Jiayi Zhou, Hantao Lou, Boyuan Chen, Donghai Hong, Xuyao Wang, Wenqi Chen, Kaile Wang, Rui Pan, Jiahao Li, Mohan Wang, Josef Dai, Tianyi Qiu, Hua Xu, Dong Li, Weipeng Chen, Jun Song, Bo Zheng, and Yaodong Yang. Align Anything: Training all-modality models to follow instructions with language feedback. arXiv preprint arXiv:2412.15838, 2024.

Minqi Jiang, Michael Dennis, Jack Parker-Holder, Jakob Foerster, Edward Grefenstette, and Tim Rocktäschel. Replay-guided adversarial environment design. In *Advances in Neural Information Processing Systems*, volume 34, pp. 1884–1897. Curran Associates, Inc., 2021a.

Minqi Jiang, Edward Grefenstette, and Tim Rocktäschel. Prioritized level replay. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4940–4950. PMLR, 18–24 Jul 2021b.

Weiyang Jin, Yuwei Niu, Jiaqi Liao, Chengqi Duan, Aoxue Li, Shenghua Gao, and Xihui Liu. SRUM: Fine-grained self-rewarding for unified multimodal models. arXiv preprint arXiv:2510.12784, 2026.

Thanh-Long V. Le, Myeongho Jeon, Kim Vu, Viet Lai, and Eunho Yang. No prompt left behind: Exploiting zero-variance prompts in LLM reinforcement learning via entropy-guided advantage shaping. In *International Conference on Learning Representations*, pp. 121956–121982, 2026.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. LLaVA-OneVision: Easy visual task transfer. *Transactions on Machine Learning Research*, 2025.

Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. SEED-Bench: Benchmarking multimodal LLMs with generative comprehension. arXiv preprint arXiv:2307.16125, 2023a.

Bohao Li, Yuying Ge, Yi Chen, Yixiao Ge, Ruimao Zhang, and Ying Shan. SEED-Bench-2-Plus: Benchmarking multimodal large language models with text-rich visual comprehension. arXiv preprint arXiv:2404.16790, 2024.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 292–305, Singapore, December 2023b. Association for Computational Linguistics.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26296–26306, June 2024.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. MMBench: Is your multi-modal model an all-around player? In *Computer Vision – ECCV 2024*, pp. 216–233, Cham, 2025. Springer Nature Switzerland.

Haoyu Lu, Wen Liu, Bo Zhang, Bingxuan Wang, Kai Dong, Bo Liu, Jingxiang Sun, Tongzheng Ren, Zhuoshu Li, Hao Yang, Yaofeng Sun, Chengqi Deng, Hanwei Xu, Zhenda Xie, and Chong Ruan. DeepSeek-VL: Towards real-world vision-language understanding. arXiv preprint arXiv:2403.05525, 2024.

Yiyang Ma, Xingchao Liu, Xiaokang Chen, Wen Liu, Chengyue Wu, Zhiyu Wu, Zizheng Pan, Zhenda Xie, Haowei Zhang, Xingkai Yu, Liang Zhao, Yisong Wang, Jiaying Liu, and Chong Ruan. JanusFlow: Harmonizing autoregression and rectified flow for unified multimodal understanding and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7739–7751, June 2025.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

Weijia Mao, Zhenheng Yang, and Mike Zheng Shou. UniRL: Self-improving unified multimodal models via supervised and reinforcement learning. arXiv preprint arXiv:2505.23380, 2025.

OpenAI, Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique P. d. O. Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with large scale deep reinforcement learning. arXiv preprint arXiv:1912.06680, 2019.

Jiadong Pan, Liang Li, Yuxin Peng, Yu-Ming Tang, Shuohuan Wang, Yu Sun, Hua Wu, Qingming Huang, and Haifeng Wang. Learning to generate via understanding: Understanding-driven intrinsic rewarding for unified multimodal models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22174–22184, June 2026.

Jack Parker-Holder, Minqi Jiang, Michael Dennis, Mikayel Samvelyan, Jakob Foerster, Edward Grefenstette, and Tim Rocktäschel. Evolving curricula with regret-based environment design. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 17473–17498. PMLR, 17–23 Jul 2022.

Qi Qin, Le Zhuo, Yi Xin, Ruoyi Du, Zhen Li, Bin Fu, Yiting Lu, Xinyue Li, Dongyang Liu, Xiangyang Zhu, Will Beddow, Erwann Millon, Victor Perez, Wenhai Wang, Yu Qiao, Bo Zhang, Xiaohong Liu, Hongsheng Li, Chang Xu, and Peng Gao. Lumina-Image 2.0: A unified and efficient image generative framework. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20031–20042, October 2025.

Lin Qiu, Hanqing Zeng, Yao Liu, Bingjun Sun, Guangdeng Liao, and Ji Liu. DUEL: Adversarial self-play for multimodal reasoning. arXiv preprint arXiv:2605.24794, 2026.

Leigang Qu, Haochuan Li, Wenjie Wang, Xiang Liu, Juncheng Li, Liqiang Nie, and Tat-Seng Chua. SILMM: Self-improving large multimodal models for compositional text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18497–18508, June 2025.

Meisam Razaviyayn, Tianjian Huang, Songtao Lu, Maher Nouiehed, Maziar Sanjabi, and Mingyi Hong. Nonconvex min-max optimization: Applications, challenges, and recent theoretical advances. *IEEE Signal Processing Magazine*, 37(5):55–66, September 2020.

Divya Saxena and Jiannong Cao. Generative adversarial networks (GANs): Challenges, solutions, and future directions. *ACM Computing Surveys*, 54(3):1–42, May 2021.

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

Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 761–769, June 2016.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, Oct 2017.

Zhaolong Su, Wang Lu, Hao Chen, Sharon Li, and Jindong Wang. UniGame: Turning a unified multimodal model into its own adversary. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 37632–37641, June 2026.

Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. arXiv preprint arXiv:2406.06525, 2024.

Zhengwei Tao, Ting-En Lin, Xiancai Chen, Hangyu Li, Yuchuan Wu, Yongbin Li, Zhi Jin, Fei Huang, Dacheng Tao, and Jingren Zhou. A survey on self-evolution of large language models. arXiv preprint arXiv:2404.14387, 2024.

Gerald Tesauro. Temporal difference learning and TD-Gammon. *Communications of the ACM*, 38 (3):58–68, March 1995.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, Austin Wang, Rob Fergus, Yann LeCun, and Saining Xie. Cambrian-1: A fully open, vision-centric exploration of multimodal LLMs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 87310–87356. Curran Associates, Inc., 2024.

Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782): 350–354, Nov 2019.

Junke Wang, Zhi Tian, Xun Wang, Xinyu Zhang, Weilin Huang, Zuxuan Wu, and Yu-Gang Jiang. SimpleAR: Pushing the frontier of autoregressive visual generation through pretraining, SFT, and RL. arXiv preprint arXiv:2504.11455, 2025.

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

Qinsi Wang, Bo Liu, Tianyi Zhou, Jing Shi, Yueqian Lin, Yiran Chen, Hai Li, Kun Wan, and Wentian Zhao. Vision-Zero: Scalable VLM self-evolution via multi-agent self-play. In *International Conference on Learning Representations*, pp. 106716–106744, 2026a.

Xin Wang, Yuwei Zhou, Bin Huang, Hong Chen, and Wenwu Zhu. Multi-modal generative AI: Multi-modal LLMs, diffusions, and the unification. *IEEE Transactions on Circuits and Systems for Video Technology*, 36(4):5621–5641, April 2026b.

Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, Yingli Zhao, Yulong Ao, Xuebin Min, Tao Li, Boya Wu, Bo Zhao, Bowen Zhang, Liangdong Wang, Guang Liu, Zheqi He, Xi Yang, Jingjing Liu, Yonghua Lin, Tiejun Huang, and Zhongyuan Wang. Emu3: Next-token prediction is all you need. arXiv preprint arXiv:2409.18869, 2024b.

Yibin Wang, Yuhang Zang, Hao Li, Cheng Jin, and Jiaqi Wang. Unified reward model for multimodal understanding and generation. arXiv preprint arXiv:2503.05236, 2026c.

Chengyue Wu, Xiaokang Chen, Zhiyu Wu, Yiyang Ma, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, Chong Ruan, and Ping Luo. Janus: Decoupling visual encoding for unified multimodal understanding and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12966–12977, June 2025a.

Penghao Wu and Saining Xie. V*: Guided visual search as a core mechanism in multimodal LLMs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13084–13094, June 2024.

Size Wu, Zhonghua Wu, Zerui Gong, Qingyi Tao, Sheng Jin, Qinyue Li, Wei Li, and Chen Change Loy. OpenUni: A simple baseline for unified multimodal understanding and generation. arXiv preprint arXiv:2505.23661, 2025b.

Enze Xie, Junsong Chen, Junyu Chen, Han Cai, Haotian Tang, Yujun Lin, Zhekai Zhang, Muyang Li, Ligeng Zhu, Yao Lu, and Song Han. SANA: Efficient high-resolution text-to-image synthesis with linear diffusion transformers. In *International Conference on Learning Representations*, pp. 20383–20407, 2025a.

Jinheng Xie, Weijia Mao, Zechen Bai, David Junhao Zhang, Weihao Wang, Kevin Qinghong Lin, Yuchao Gu, Zhijie Chen, Zhenheng Yang, and Mike Zheng Shou. Show-o: One single transformer to unify multimodal understanding and generation. In *International Conference on Learning Representations*, pp. 28240–28264, 2025b.

Jinheng Xie, Zhenheng Yang, and Mike Zheng Shou. Show-o2: Improved native unified multimodal models. In *Advances in Neural Information Processing Systems*, volume 38, pp. 47490–47518. Curran Associates, Inc., 2025c.

Shenghao Xie, Wenqiang Zu, Mingyang Zhao, Duo Su, Shilong Liu, Ruohua Shi, Guoqi Li, Shanghang Zhang, and Lei Ma. Towards unifying understanding and generation in the era of vision foundation models: A survey from the autoregression perspective. arXiv preprint arXiv:2410.22217, 2024.

Haoyan Yang, Mario Xerri, Solha Park, Huajian Zhang, Yiyang Feng, Sai Akhil Kogilathota, and Jiawei Zhou. Self-improvement of large language models: A technical overview and future outlook. *Transactions on Machine Learning Research*, 2026. Survey Certification.

Ling Yang, Xinchen Zhang, Ye Tian, Shiyi Zhang, Chenming Shang, Minghao Xu, Wentao Zhang, and Bin Cui. HermesFlow: Seamlessly closing the gap in multimodal understanding and generation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 62248–62272. Curran Associates, Inc., 2025.

Ziyu Ye, Rishabh Agarwal, Tianqi Liu, Rishabh Joshi, Sarmishta Velury, Quoc V. Le, Qijun Tan, and Yuan Liu. Scalable reinforcement post-training beyond static human prompts: Evolving alignment via asymmetric self-play. arXiv preprint arXiv:2411.00062, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, volume 38, pp. 113222–113244. Curran Associates, Inc., 2025.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11975–11986, October 2023.

Andrew Zhao, Yiran Wu, Tong Wu, Quentin Xu, Yang Yue, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute Zero: Reinforced self-play reasoning with zero data. In *Advances in Neural Information Processing Systems*, volume 38, pp. 105816–105879. Curran Associates, Inc., 2025.

Shanshan Zhao, Xinjie Zhang, Jintao Guo, Jiakui Hu, Lunhao Duan, Minghao Fu, Yong Xien Chng, Guo-Hua Wang, Qing-Guo Chen, Zhao Xu, Weihua Luo, and Kaifu Zhang. Unified multimodal understanding and generation models: Advances, challenges, and opportunities. arXiv preprint arXiv:2505.02567, 2026.

APPENDIX

This appendix supplements the main text in three parts. Appendix A compares MATE with the closest post-training methods for UMMs. Appendix B gives the full training procedure, the two directions of a round, and the GRPO update. Appendix C specifies the scoring functions and hyperparameters and reports additional variants of MATE, a second example of evolving training inputs, and details of the round-trip evaluation.

A POSITIONING AGAINST PRIOR WORK

Table 3 compares MATE with the post-training methods for UMMs discussed in Section 2, which are the cooperative self-training methods and UniGame. The column headed *Other branch supplies* states what one branch contributes to the training of the other. In the cooperative methods, this branch scores the outputs of the other through the likelihood that it assigns to the original input, the correctness of its answers to questions derived from the prompt, or a rating that it writes. In UniGame and MATE it supplies challenges instead, and the scores in MATE come from fixed scoring functions (Appendix C). *Tasks optimized* lists the tasks with training objectives, where G and U denote generation and understanding; shared parameters can affect both capabilities. *Training inputs* states where the training inputs come from. A corpus is fixed before training, constructed prompts are built so that the answers to their questions are known, and model-written prompts are produced by the model under training. UniGame adds a buffer that replays hard examples, whereas MATE rebuilds its input pools every epoch from the outputs selected in the previous one. *Hard cases selected for* lists the tasks whose training inputs are selected by solver difficulty, and *Added module* names any trainable module that a method adds to the model.

Table 3: Comparison of MATE with the closest post-training methods for UMMs. Hard-case selection refers to choosing training inputs by the difficulty they pose to the task being optimized.

Method	Other branch supplies	Tasks optimized	Training inputs	Hard cases selected for	Added module
SUDER (Hong et al., 2025)	likelihood	G, U	corpus	–	–
GvU (Pan et al., 2026)	likelihood	G	corpus	–	–
UniRL (Mao et al., 2025)	answers	G, U	constructed prompts	–	–
SILMM (Qu et al., 2025)	answers	G	model-written prompts	–	–
HermesFlow (Yang et al., 2025)	answers	G, U	corpus	–	–
SRUM (Jin et al., 2026)	ratings	G	corpus	–	–
UniCorn (Fang et al., 2026)	ratings	G, U	model-written prompts	–	–
UniGame (Su et al., 2026)	challenges	U	corpus, replay buffer	U	perturber
MATE (ours)	challenges	G, U	evolving pools	G, U	–

B ADDITIONAL DETAILS ON THE ALGORITHM

Algorithm 1 is the full procedure of Section 3. Algorithms 2 and 3 give the round of Section 3.2 in its two directions. In FORWARDLOOP the round starts from a source text, the generator is the challenger, and the understanding branch is the solver. REVERSELOOP starts from a source image and exchanges the two roles, which is the case formulated in Equations 1 to 3. Each procedure returns the selected candidate together with its group of M rollouts, and Algorithm 1 updates the solver with GRPO on the returned groups.

Algorithm 1 MATE: one training run

Require: input pools $\mathcal{P}_0^p$ and $\mathcal{P}_0^I$ drawn from $\mathcal{D}$, UMM $\boldsymbol{\theta}$, epoch count E, floor ϵ
1: **for** $k = 0$ to $E - 1$ **do** ▷ update both branches in each epoch
2: For each $\boldsymbol{p} \in \mathcal{P}_k^p$: $(\boldsymbol{I}_{\boldsymbol{p}}^\star,\ g_{\boldsymbol{p}}) \leftarrow \textsc{ForwardLoop}(\boldsymbol{p}, \boldsymbol{\theta})$ ▷ hard images for $\mathsf{U}_{\boldsymbol{\theta}}$
3: For each $\boldsymbol{I} \in \mathcal{P}_k^I$: $(\boldsymbol{p}_{\boldsymbol{I}}^\star,\ g_{\boldsymbol{I}}) \leftarrow \textsc{ReverseLoop}(\boldsymbol{I}, \boldsymbol{\theta})$ ▷ hard descriptions for $\mathsf{G}_{\boldsymbol{\theta}}$
4: Update $\boldsymbol{\theta}$ with GRPO on the groups $\{g_{\boldsymbol{p}}\}$ whose reward standard deviation is at least ϵ, then likewise on $\{g_{\boldsymbol{I}}\}$
5: $\mathcal{P}_{k+1}^I \leftarrow \{\boldsymbol{I}_{\boldsymbol{p}}^\star : \boldsymbol{p} \in \mathcal{P}_k^p\}, \quad \mathcal{P}_{k+1}^p \leftarrow \{\boldsymbol{p}_{\boldsymbol{I}}^\star : \boldsymbol{I} \in \mathcal{P}_k^I\}$ ▷ inputs of the other loop
6: **end for**
7: **return** updated model $\boldsymbol{\theta}$

Let $r_1, \ldots, r_M$ denote the rollout rewards of a returned group, $\bar{r}$ their mean, and σ their standard deviation. The group-relative advantage of rollout m is

$$A_m = \frac{r_m - \bar{r}}{\sigma}. \tag{5}$$

The update then follows the clipped GRPO objective of Shao et al. (2024) with these advantages and a KL penalty toward a reference policy, and Table 4 lists the clipping range and the penalty weight. A group whose standard deviation falls below the floor ϵ is excluded from the update, since its rewards are nearly identical and their standardized advantages carry no signal. DAPO (Yu et al., 2025) likewise filters groups with no reward variation, whereas other work reshapes their advantages (Le et al., 2026) or returns their queries to the same policy (Coelho et al., 2026). In MATE the selected candidate of an excluded group still enters the next epoch's input pool as a source for the other branch, so the floor does not change challenge selection. Rollouts from unselected candidates are used only to estimate their difficulty. Training on the groups of all admitted candidates instead is evaluated in Table 5.

Algorithm 2 ForwardLoop: selecting challenging images for the understanding branch

Require: source text $\boldsymbol{p}$, UMM $\boldsymbol{\theta}$, scoring functions R_G and R_U, consistency measure C_U, weight λ, threshold τ, candidate count N, rollout count M
1: **for** $n = 1$ to N **do**
2: Sample candidate image $\boldsymbol{I}_n \sim \mathsf{G}_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{p})$
3: Compute admission score $\mathsf{R}_\mathsf{G}(\boldsymbol{I}_n, \boldsymbol{p})$
4: **end for**
5: Form admitted set $\mathcal{F} = \{n : \mathsf{R}_\mathsf{G}(\boldsymbol{I}_n, \boldsymbol{p}) > \tau\}$
6: **if** $\mathcal{F} = \emptyset$ **then**
7: $\mathcal{F} \leftarrow \{\arg\max_n \mathsf{R}_\mathsf{G}(\boldsymbol{I}_n, \boldsymbol{p})\}$
8: **end if**
9: **for** each $n \in \mathcal{F}$ **do**
10: **for** $m = 1$ to M **do**
11: Sample description $\hat{\boldsymbol{p}}_{n,m} \sim \mathsf{U}_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{I}_n)$
12: $r_{n,m} \leftarrow \mathsf{R}_\mathsf{U}(\hat{\boldsymbol{p}}_{n,m}, \boldsymbol{I}_n) + \lambda\, \mathsf{C}_\mathsf{U}(\hat{\boldsymbol{p}}_{n,m}, \boldsymbol{p})$
13: **end for**
14: $\bar{r}_n \leftarrow \frac{1}{M} \sum_{m=1}^{M} r_{n,m}$ ▷ mean solver score on candidate $\boldsymbol{I}_n$
15: **end for**
16: $n^\star \leftarrow \arg\min_{n \in \mathcal{F}} \bar{r}_n$
17: **return** selected candidate $\boldsymbol{I}_{n^\star}$ and its group $\{(\hat{\boldsymbol{p}}_{n^\star,m}, r_{n^\star,m})\}_{m=1}^{M}$

Algorithm 3 REVERSELOOP: selecting challenging descriptions for the generator

Require: source image $\boldsymbol{I}$, UMM $\boldsymbol{\theta}$, scoring functions R_G and R_U, consistency measure C_G, weight λ, threshold τ, candidate count N, rollout count M

for $n = 1$ to N **do**
  Sample candidate description $\boldsymbol{p}_n \sim \mathsf{U}_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{I})$
  Compute admission score $\mathsf{R}_\mathsf{U}(\boldsymbol{p}_n, \boldsymbol{I})$
end for
Form admitted set $\mathcal{F} = \{n : \mathsf{R}_\mathsf{U}(\boldsymbol{p}_n, \boldsymbol{I}) > \tau\}$
if $\mathcal{F} = \emptyset$ **then**
  $\mathcal{F} \leftarrow \{\arg\max_n \mathsf{R}_\mathsf{U}(\boldsymbol{p}_n, \boldsymbol{I})\}$
end if
for each $n \in \mathcal{F}$ **do**
  for $m = 1$ to M **do**
    Sample image $\hat{\boldsymbol{I}}_{n,m} \sim \mathsf{G}_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{p}_n)$
    $r_{n,m} \leftarrow \mathsf{R}_\mathsf{G}(\hat{\boldsymbol{I}}_{n,m}, \boldsymbol{p}_n) + \lambda\, \mathsf{C}_\mathsf{G}(\hat{\boldsymbol{I}}_{n,m}, \boldsymbol{I})$
  end for
  $\bar{r}_n \leftarrow \frac{1}{M} \sum_{m=1}^{M} r_{n,m}$ ▷ mean solver score on candidate $\boldsymbol{p}_n$
end for
$n^\star \leftarrow \arg\min_{n \in \mathcal{F}} \bar{r}_n$
return selected candidate $\boldsymbol{p}_{n^\star}$ and its group $\{(\hat{\boldsymbol{I}}_{n^\star,m}, r_{n^\star,m})\}_{m=1}^{M}$

C EXPERIMENTAL DETAILS

Scoring functions. UnifiedReward (Wang et al., 2026c) implements both scoring functions (Section 4.1); we use the released checkpoint `UnifiedReward-qwen-7b`, built on Qwen2.5-VL-7B-Instruct, with greedy decoding. Its pointwise image-against-text scores define R_G on a 1–5 scale, and its pointwise text-against-image scores define R_U on a 0–100 scale. We use the released pointwise prompts unchanged across all runs and rescale both scores to $[0, 1]$ before they are used. The same scores decide admission against τ. UnifiedReward takes no part in evaluation.

Hyperparameters and parameter counts. Table 4 lists the training hyperparameters and the evaluation settings of Section 4. The parameter counts in Tables 1 and 2 are the language-model parameters for UMMs and understanding-only models and the total parameters for generation-only models, as reported in their respective papers.

Table 4: Training hyperparameters and evaluation settings.

Training			
Candidates per input N	4	Learning rate	10^{-6}
Rollouts per candidate M	8	GRPO clipping range	0.2
Admission threshold τ	0.5	Reference penalty	0.01 (low-variance KL)
Standard-deviation floor ϵ	10^{-6}	Trainable parameters	all (2.1B)
Epochs	5	Hardware	6× NVIDIA B200
Effective batch size	48 (24 inputs per step)	Consistency weight λ	0.1
Evaluation			
Image sampling	384 px, guidance weight 5, temperature 1, 4 images per prompt		
Text decoding	greedy		

Additional variants. Table 5 reports the understanding and generation averages of three additional variants of MATE, together with Janus-Pro-1B and MATE for reference. Both averages are defined in Section 4 and are computed before rounding. The first variant removes the consistency term, setting $\lambda = 0$ in Equation 3. Without it, both averages remain within 0.2 points of those of MATE, so at this scale the term has little effect on them. The next variant trains on the groups of all admitted candidates instead of the selected group alone, which gives up to N groups per source rather than one. Its averages are also within 0.3 points of those of MATE. The last variant runs MATE on two separate networks, using Qwen2-VL-2B (Wang et al., 2024a) for understanding and SimpleAR-0.5B (Wang et al., 2025) for generation. The pair has 2.8B parameters in total, compared with 2.1B for the

complete Janus-Pro-1B model. Relative to its own base models, the pair gains 2.1 points on the generation average and loses 0.9 points on the understanding average. MATE improves both averages with a shared backbone, including a 0.7-point gain in understanding. This comparison is consistent with a benefit from shared parameters, although the systems also differ in their base models and total parameter counts.

Table 5: Additional variants of MATE, with Janus-Pro-1B and MATE for reference.

	Und. Avg.↑	Gen. Avg.↑
Janus-Pro-1B	56.9	78.3
MATE	57.6	80.3
w/o consistency term	57.5	80.5
All admitted candidates	57.5	80.6
Two separate networks	66.3	64.1
Δ vs. own base models	-0.9	$+2.1$

Evolving training examples. Figure 7 follows a second original dataset pair through five epochs of MATE training, complementing the example in Figure 5.

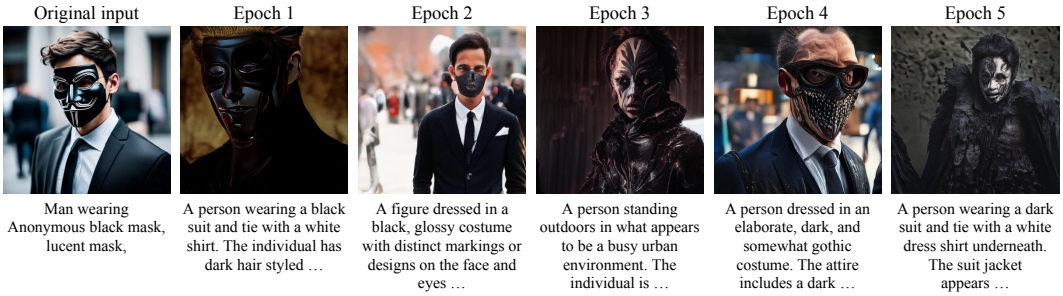


Figure 7: **A second evolving training input under MATE,** displayed as in Figure 5. Descriptions are shortened for display.

A complete round. Figure 8 shows every candidate, attempt, and score of one training pair in one epoch, laid out stage by stage as in Figure 2(a) for both directions of the round; the two selected outputs become the pair's inputs in the next epoch.

Round-trip evaluation. This paragraph specifies the encoders and two details of the evaluation in Section 4.3. The image-side similarity uses the SigLIP-L/16-384 vision encoder (Zhai et al., 2023) bundled with Janus-Pro-1B (Chen et al., 2025): each image is resized to 384 pixels, the 576 patch embeddings are averaged, and the result is ℓ_2-normalized. The text-side similarity uses the text encoder of the released `siglip-large-patch16-384` checkpoint (Zhai et al., 2023). It compares the first description $\boldsymbol{p}_1$ with the description $\boldsymbol{p}_k$ of the image produced in round $k-1$, so it is defined from $k=2$. A description longer than the 64-token window of the SigLIP text encoder is split into windows, and the window embeddings are averaged. The 32 held-out samples are the first 32 pairs of the Align-Anything test split; descriptions are decoded greedily with at most 256 tokens, and images are sampled with the settings of Table 4.

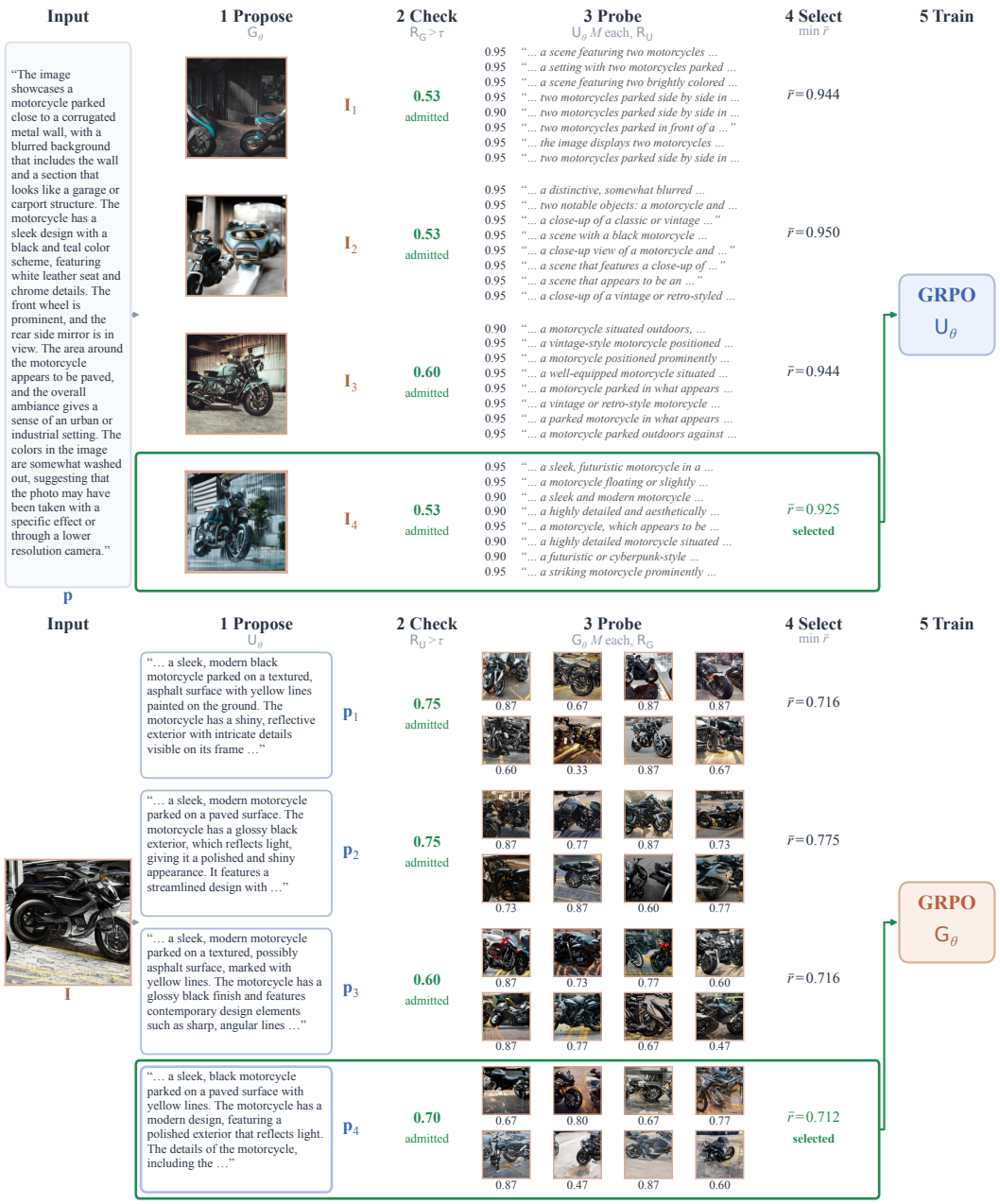


Figure 8: **One complete round of MATE, in the format of Figure 2(a).** All candidates, attempts, and scores of one training pair in one epoch (pair 4, epoch 2 of the main run). Upper row: from the source description $\boldsymbol{p}$, the generation branch proposes $N = 4$ images $\boldsymbol{I}_1, \ldots, \boldsymbol{I}_4$; each is checked against $\boldsymbol{p}$ (R_G, admission threshold $\tau = 0.5$); the understanding branch writes $M = 8$ descriptions of each admitted image, scored by R_U (the eight scores and one example are shown); the image with the lowest mean score $\bar{r}$ is selected and its eight attempts train $\mathsf{U}_{\boldsymbol{\theta}}$ with GRPO. Lower row: from the source image $\boldsymbol{I}$, the understanding branch proposes $N = 4$ descriptions $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_4$, checked against $\boldsymbol{I}$ (R_U); the generation branch produces $M = 8$ images from each admitted description, scored by R_G; the description with the lowest $\bar{r}$ is selected and its attempts train $\mathsf{G}_{\boldsymbol{\theta}}$. The two selected outputs become this pair's inputs in the next epoch. Descriptions are shortened for display.