UniGame: Turning a Unified Multimodal Model Into Its Own Adversary

Supplementary Material

Appendix Contents

A. Algorithm Details

The complete training algorithm of UniGame is shown in Algorithm 1.

Algorithm 1 UniGame

Initialize θ_U (understanding) and θ_C (Perturber);
for each training step $t = 1, 2, \ldots$ **do**
  Sample minibatch $\{(\mathbf{x}_i, q_i, a_i)\}_{i=1}^{M} \sim \mathcal{D}$ and encode $\mathbf{z}_i = \mathrm{Proj}(\mathrm{Enc}(\mathbf{x}_i))$
  Challenge step (update C):
  Compute perturbations $\boldsymbol{\delta}_i = C(\mathbf{z}_i; \theta_C)$ and perturbed tokens $\tilde{\mathbf{z}}_i = \mathbf{z}_i + \boldsymbol{\delta}_i$ with $\|\boldsymbol{\delta}_i\| \le \varepsilon_{\max}$
  Decode candidates $\tilde{\mathbf{x}}_i = G(\tilde{\mathbf{z}}_i)$
  Compute $\mathcal{L}_C(\theta_C; \theta_U)$ as in Eq. (7)
  Update $\theta_C \leftarrow \theta_C + \eta_C \nabla_{\theta_C} \mathcal{L}_C$
  if $t \bmod m = 0$ **then**
    Compute scores H_j and keep candidates passing CLIP threshold τ and push hard examples into $\mathcal{B}$ via Eq. (4)
  end if
  Understand step (update U):
  Construct mixed batch: clean samples $(\mathbf{z}_i, q_i, a_i)$, and hard samples $(\hat{\mathbf{z}}_j, \hat{q}_j, \hat{a}_j)$ drawn from $\mathcal{B}$
  Compute $\mathcal{L}_U(\theta_U)$ on the mixed batch as in Eq. (6)
  Update $\theta_U \leftarrow \theta_U - \eta_U \nabla_{\theta_U} \mathcal{L}_U$
end for

B. Training Details

B.1. Training and Testing Data

Data volumes. Unless otherwise noted, we follow the official training/evaluation splits and report results on the standard benchmarks. Training uses **VQAV2 train-split** [11] is a large-scale visual question answering benchmark (hundreds of thousands of image–question pairs) collected from MS-COCO images with crowd-sourced free-form answers; it emphasizes grounded visual reasoning under natural images. **CC3M** [35] (training only) is a large web-scale image–caption corpus (∼3M pairs in the full set); we use a filtered *subset* of 100k as text–image supervision for the generative branch.

Benchmarks. We briefly introduce the benchmarks:

- **VQAv2 test-dev** [11]: the official VQAv2 test-dev split contains **104 000** questions; evaluation is via the online server.[5]
- **MMMU** [45]: a college-level, multi-discipline benchmark with **11 500** questions in total (we report on the official test set).
- **POPE** [17]: object-hallucination evaluation with a balanced, image-grounded design; the **test split has 9000** QA pairs.
- **MMBench** [20]: curated multiple-choice suite; **dev 1164** and **test 1784** questions (4:6 split of ∼3K).
- **GenEval** [8]: object/layout/attribute–focused T2I evaluation with **553** prompts (reference-free automatic checks).
- **UnifiedBench** [43]: unification score via caption→reconstruction; Protocol-1 uses **100 source images**.
- **WISE** [23]: knowledge-informed T2I evaluation with **1000** structured prompts across 25 subdomains.
- **NaturalBench** [14]: vision-centric VQA with natural adversarial samples, ∼10 000 human-verified image–question pairs (2500 *groups* under the 2-image×2-question protocol), scored by G-Acc.
- **AdVQA** [16]: human-in-the-loop adversarial VQA; total size reported as ∼**46 807** examples (commonly used splits include ∼**5123** val / ∼**23 399** test).

B.2. Hyperparameter Details

Optimization details. UniGame is like the current UMMs post-training, is an end-to-end method and involves decoding images in each batch, to balance performance and cost. Our optimizations are as followed. We use AdamW optimizers with learning rates for Generation (`gen_lr`) and Understanding (`und_lr`). We conduct extensive ablation on the learning rate ratio between these two components (detailed in Appendix C and Table 8), ultimately finding that a ratio of approximately 250 achieves optimal performance (`gen_lr`$= 5\times10^{-3}$, `und_lr`$= 2\times10^{-5}$).

We implement mixed precision for training, given that Uni-Game only learned and uses small-norm perturbation, insufficient numerical precision can quantize away the perturbation's gradients and wash out all the supervision.

[5] Counts from the official VQA site; see also recent reports confirming 104K for test-dev.

We vary the Generation and understanding update ratio in {1:1, 1:5, 1:10}. We performed a precision ablation comparing `fp16-all`, `bf16-all`, `tf32-enabled`, `fp32-all`, `fp16(G)+fp32(loss)`, and `bf16(G/D)+fp32(loss)`, and found that our final choice—computing the perturbation update, regularizer, and losses in `float32` while running the remaining forward/backward in `bfloat16`—consistently achieved the best stability–efficiency trade-off and the highest robust accuracy. We force all computations that determine the perturbation and its supervision to `float32`. Gradient norms and per-role clipping are also applied in FP32, and optimizer states remain FP32 (AdamW default). All other forward/backward passes (vision tower, diffusion decoder, and LLM blocks) run under `bfloat16 autocast` for throughput. This preserves the perturbation signal while retaining the speed benefits of mixed precision.

B.3. Perturber

Network architecture of C. We implement the perturber C as a lightweight three-layer MLP that operates on each fused visual token after the language model. The first two layers have the same width as the UMM hidden size and apply non-linear transformations that refine the token representation and extract a direction in the shared visual-token space. The third layer acts as a direction head, mapping the hidden representation back to the token space and indicating along which semantic direction each token should be pushed to maximally challenge the understanding branch. In parallel, C maintains a single learnable scalar gate ε, shared across tokens and constrained within the perturbation budget $[0, \varepsilon_{\max}]$, which controls the overall perturbation strength. In this way, one part of C is responsible for discovering semantically adversarial directions, while the scalar gate ε controls how strongly these directions are applied, keeping the module compact (with $|\theta_C| \ll \min(|\theta_U|, |\theta_G|)$) yet able to generate small but semantically meaningful adversarial perturbations.

B.4. Hard Samples

UniGame added a hard sampler buffer to select only the challenging adversarial samples for training. Figure 6 shows some challenging examples in our experiments.

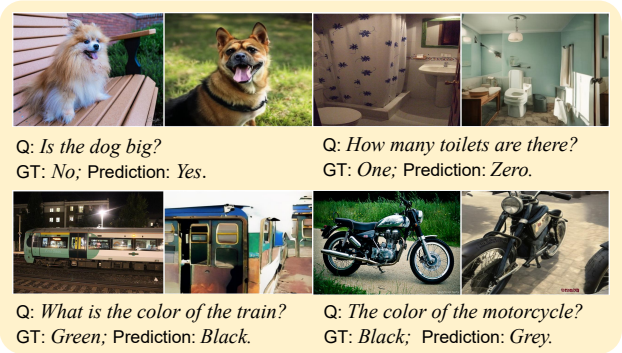

Figure 6. Cases are drawn from the hard-sample buffer and represent failure cases that successfully challenged the model.

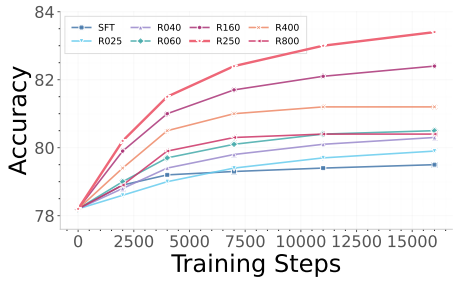

Figure 7. Training dynamics analysis. VQA accuracy evolution across different adversarial ratios, with best achieving optimal performance at 83.4%.

C. Detailed Experimental Results

C.1. Additional Backbone Validation

To validate the generality of UniGame beyond Janus-Pro-7B, we applied it to two additional UMM backbones: BLIP-3o [1] (diffusion-based) and Chameleon [36] (auto-regressive), both trained on a 0.4 split of the VQAv2 training set under-matched settings. All backbones show positive consistency gains: BLIP-3o (+2.5, MMMU 51.2, GenEval 0.54), and Chameleon (+2.2, VQAv2 40.2 with +1.7 gain, MMMU 24.0, GenEval 0.40). These results confirm that UniGame generalizes across different UMM architectures, including both auto-regressive and diffusion-based designs.

C.2. Learning Rate Ratio Ablation

To determine the optimal balance between the generation and understanding branches, we conduct an extensive sweep of learning rate ratios. Table 8 lists the complete set of configurations tested.

Table 8. Learning-rate configurations for the adversarial ratio sweep. Each row (ID R*xxx*) specifies a pair of learning rates for the generation (`gen_lr`) and understanding module (`und_lr`); the last column reports their ratio Gen/Und. For example, R250 corresponds to `gen_lr`$= 5 \times 10^{-3}$ and `und_lr`$= 2 \times 10^{-5}$, i.e., a 250:1 ratio. These IDs (R025–R800) are used in Fig. 7(b) to plot validation performance as a function of the adversarial ratio.

ID	gen_lr	und_lr	Gen/Und
R025	1.6×10^{-3}	6.3×10^{-5}	≈ 25.4
R040	2×10^{-3}	5×10^{-5}	40
R060	2.4×10^{-3}	4.1×10^{-5}	≈ 58.5
R100	3.2×10^{-3}	3.2×10^{-5}	100
R160	4×10^{-3}	2.5×10^{-5}	160
R250	5×10^{-3}	2×10^{-5}	250
R400	6.3×10^{-3}	1.6×10^{-5}	≈ 394
R600	7.7×10^{-3}	1.3×10^{-5}	≈ 592
R800	8.9×10^{-3}	1.1×10^{-5}	≈ 809

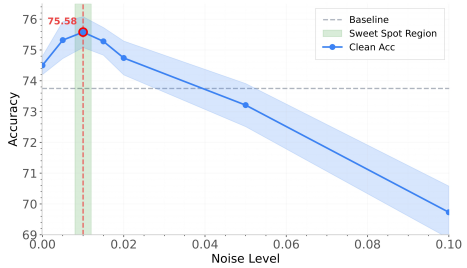

Figure 9. Perturbation Sweetspot

C.3. Motivation Experiments

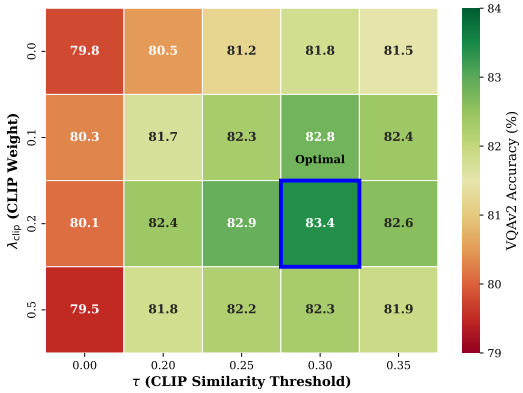

Figure 8. heatmap Ablation study on CLIP constraint configurations. We report VQAv2 accuracy for different combinations of CLIP weight and CLIP similarity threshold

To find an Optimal noise level, we inject i.i.d. Gaussian noise into the projected visual tokens with $\sigma \in \{0, 0.005, 0.01, 0.015, 0.02, 0.05, 0.1\}$. We observe a *sweet spot* near $\sigma \approx 0.01$ where VQAv2 soft accuracy slightly increases ($74.50 \rightarrow 75.58$) before degrading at larger noise, see in Figure 9. This indicates that small, structured embedding perturbations can beneficially modulate the shared representation.

D. Robustness Results

The details results on OOD and adversarial robustness are shown in Table 9, indicating that UniGame significantly improves the robustness of the models.

Table 9. Results for OOD and adversarial robustness.

Model	NaturalBench	AdVQA
Janus-Pro	73.8	34.2
+SFT	73.9	36.4
+Ours	78.6	40.4

E. Details on Case Study

E.1. Case Study on Understanding Tasks

We offer more interpretations to Figure 5.

- **Object counting** (C1): The baseline model fails to accurately count objects in cluttered scenes, often confusing similar-looking items or missing partially visible objects. After UniGame training, the model correctly identifies the precise count, demonstrating improved fine-grained visual attention.
- **Object interaction** (C2): Understanding relational semantics between objects (e.g., "person holding umbrella" vs. "umbrella next to person") requires compositional reasoning. The baseline misinterprets spatial relationships, while UniGame correctly recognizes the interaction pattern.
- **Spatial relation and location** (C3): Queries about relative positions (e.g., "left of", "behind") expose fragile spatial understanding in the baseline. UniGame's adversarial training—which systematically perturbs spatial layouts during decoding—hardens the model against such failures.
- **Crowd object detection** (C4): dense and overlapping objects in crowded scenes challenge both localization and recognition. The baseline produces vague or incorrect answers, whereas UniGame maintains accuracy by learning from decoded adversarial samples that emphasize occlusion and clutter.

These qualitative improvements align with our quantitative gains, confirming that UniGame systematically addresses decision-critical reasoning failures rather than merely fitting to benchmark statistics.

In addition, we also present detailed analysis to the open-ended understanding tasks:

- **Open-ended understanding.** As illustrated in Figure 5a, UniGame produces more fine-grained and visually grounded captions than the baseline. The model not only recognizes the overall scene (e.g., pizza, street sign, animals) but also reliably captures details such as a missing pizza slice, vegetables on display at a farmers market, a cat sitting on a windowsill looking out the window, or two sheep wearing coats standing in a field. These examples show that adversarial self-play improves open-ended descriptions by encouraging the model to focus on decision-critical visual evidence rather than hallucinated or overly generic content.

E.2. Case Study on Generation Tasks

We offer more detailed explanation of the text-to-image generations in Figure 5b.

- On the synthetic shapes example, the baseline model already produces plausible objects but often violates fine-grained layout constraints (e.g., incorrect left/right ordering or cube–sphere counts), whereas UniGame yields images that respect the specified 2×2 red cube stack, the correct number of blue spheres, and the spatial relations such as "on the left / on the right" and "between".
- In the "broccoli in a glass bowl" example, UniGame more faithfully binds multiple attributes—three pieces of broccoli, two carrots on the side, and a clearly visible red sticker with the number "5" attached to the bowl—demonstrating stronger compositional control.
- For the Grand Canyon scene, the baseline sometimes collapses the layered rock formations into a flatter composition, while UniGame better preserves depth and lighting that match the prompt description.
- Finally, for the "blue-eyed Siamese cat sitting on a green velvet armchair", UniGame produces a sharper Siamese appearance and a more coherent green velvet texture, indicating that self-play training can improve both semantic alignment and visual fidelity.

F. Convergence and Hyperparameter Analysis

F.1. Convergence

Convergence of the minimax training. The minmax setup raises the practical question: when does the game converge and what schedules keep it stable? In our setup, only the Perturber C and LoRA adapters on the understanding branch U are trainable; due to U's larger capacity, it can dominate and degrade the generation module. We restore stability by giving C a higher learning rate and using short, interleaved updates. We conducted an extensive sweep of the Generation/Understanding update ratio in Table 8, shows $\texttt{gen\_lr} = 5 \times 10^{-3}$, $\texttt{und\_lr} = 2 \times 10^{-5}$, provides the best clean–robust trade-off; prolonged generation phases saturate the attack success rate (ASR) before U adapts and induce catch-up oscillations (see Figure 13 Figure 11). Conversely, when the generation overpowers U, decoded candidates drift off-manifold and hurt clean accuracy. Thus, *balance progression speeds*: (i) use a slightly larger learning rate for C than for U's adapters, and (ii) prefer short alternations over long unilateral bursts. Full grids, curves, and ablations are shown in Section C. **Perturbation budget.** The budget constraint $\varepsilon_{\max}$ controls the perturbation magnitude in the token space. The results in Appendix C show a sweetspot that inverted U-shaped performance curve Figure 9, setting $\varepsilon_{\max}$ too small (e.g., 0.005) produces weak perturbations that fail to expose critical reasoning failures, yielding limited robustness gains ($+1.7\%$ on NaturalBench).

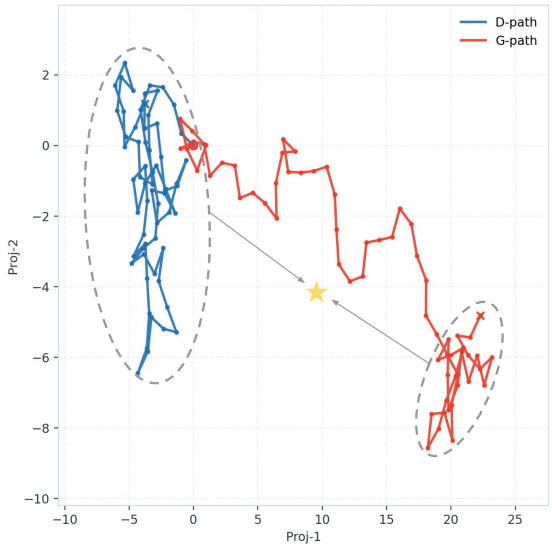

Figure 10. The best result of all of our runs, optimization path are projected to a two dimension axis.

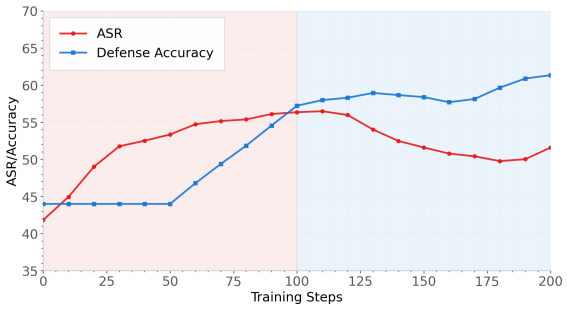

Figure 11. Self-play dynamics between the generation and the understanding . The two branches alternately dominate the training objective, exhibiting a stable tug-of-war behavior.

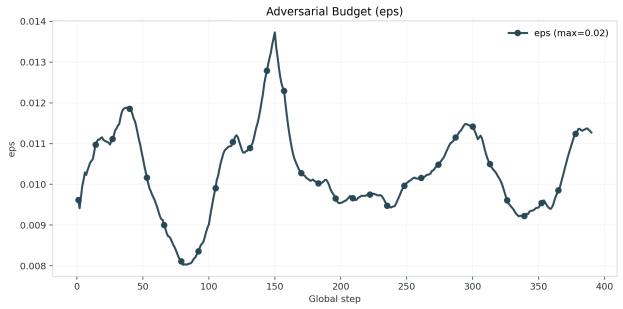

Figure 12. Perturebation Budget

F.2. Hyperparameter Sensitivity Analysis

Unless otherwise noted, we fix the perturbation budget to $\delta = \varepsilon_{\max} = 0.02$ in all main experiments, which we found to provide a good clean–robust trade-off after sweeping $\delta \in \{0.005, 0.01, 0.015, 0.02, 0.05, 0.10\}$ Figure 12. For hard-example mining, we define the hardness score H as the cross-entropy loss of the understanding branch on decoded candidates plus a CLIP-based hinge term, and select hard samples using a quantile-based threshold: the buffer threshold τ is set to the 60-th percentile of H within each mining batch, while additionally enforcing a minimum text–image CLIP similarity of 0.6 to filter out semantically off-manifold generations. The trade-off coefficient β in Eq. (6), which weights the contribution of buffer samples relative to clean examples, is set to $\beta = 0.5$ by default so that roughly half of the understanding gradient comes from adversarial or hard instances; we observed that UniGame is numerically stable for a broad range of $\beta \in [0.3, 1.0]$. The hard-sample replay buffer stores up to 50 decoded images ranked by H. We deliberately keep the capacity moderate, as substantially larger buffers (e.g., $\gg 10^4$ entries) would store many full-resolution decoded images and quickly lead to a steep increase in GPU and host memory usage, without providing noticeable additional benefits in practice.

G. Theoretical Insights

In this section, we provide preliminary theoretical justification for why the proposed minimax self-play procedure improves (i) the stability of the understanding branch, (ii) convergence of the alternating optimization, and (iii) coverage of the shared generative manifold. The analysis is intentionally model-agnostic and applies to a broad class of unified multimodal architectures.

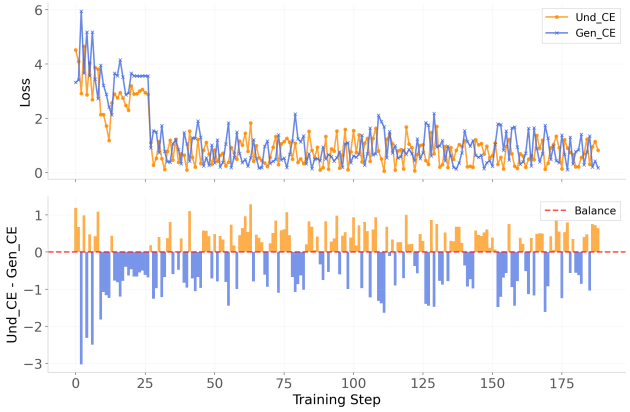

Figure 13. Dominance timeline. The trajectory alternates between understanding and generation phases, illustrating a stable tug-of-war rather than collapse to either side during training.

G.1. Convergence of the Minimax Self-Play Dynamics

Recall the UniGame objective

$$\min_{\theta_U} \max_{\theta_C} \mathcal{L}(\theta_U, \theta_C) = \mathbb{E}\big[\ell_U(\theta_U)\big] + \lambda\, \mathbb{E}\big[\ell_C(\theta_C; \theta_U)\big], \tag{8}$$

where the perturber maximizes the understanding loss while the understanding head minimizes both clean and adversarial losses, subject to a bounded perturbation $\|\delta\| \le \varepsilon_{\max}$ in the shared token space. In this subsection, we analyze an *idealized* version of this minimax problem to provide theoretical intuition, rather than a full convergence proof for the actual deep network implementation.

Assumption 1 (Lipschitz continuity). The understanding loss $\ell_U(a \mid z, q)$ is L-Lipschitz continuous in the token embedding z and continuously differentiable in θ_U.

Assumption 2 (Bounded perturbation set and parameter domain). The perturber operates within a compact, convex set

$$\mathcal{D} = \{\delta : \|\delta\| \le \varepsilon_{\max}\}. \tag{9}$$

Moreover, the parameter sets Θ_U and Θ_C for θ_U and θ_C are assumed to be compact and convex.

Assumption 3 (Local nonconvex–concave structure). For any fixed $\theta_U \in \Theta_U$, the function $\theta_C \mapsto \mathcal{L}(\theta_U, \theta_C)$ is (locally) concave on Θ_C in a neighborhood of the stationary points of interest. Equivalently, the game is nonconvex in θ_U and (locally) concave in θ_C around those points.

Proposition 1 (First-order stationary point and stability). Under Assumptions 1–3, the minimax problem in

Eq. (8) admits at least one first-order stationary point (θ_U^*, θ_C^*), i.e.,

$$\nabla_{\theta_U} \mathcal{L}(\theta_U^*, \theta_C^*) = 0, \qquad \nabla_{\theta_C} \mathcal{L}(\theta_U^*, \theta_C^*) = 0.$$

Moreover, for sufficiently small learning rates (η_U, η_C), gradient descent–ascent generates a bounded sequence and converges to a neighborhood of a first-order stationary point of $\mathcal{L}$.

Sketch of proof. By Assumption 2, the feasible set in $(\theta_U, \theta_C, \delta)$ is compact and convex, so a minimax solution and hence a first-order stationary point exist. Assumption 1 guarantees that the loss is smooth in θ_U, and Assumption 3 provides a local nonconvex–concave structure: for each fixed θ_U, the objective is (locally) concave in θ_C. Under such smooth nonconvex–concave conditions, standard results for two-player minimax optimization show that gradient descent–ascent with sufficiently small step sizes (η_U, η_C) generates a bounded sequence and converges to an $\mathcal{O}(\eta_U + \eta_C)$ neighborhood of a first-order stationary point of $\mathcal{L}$.

Implication. These assumptions suggest that the adversarial self-play dynamics are stable and tend not to diverge, even though the perturber and understanding branches pursue opposing objectives.

G.2. Robustness Improvement via Worst-Case Regularization

For a fixed sample z from the shared representation space, the creator seeks a worst-case perturbation

$$\max_{\|\delta\| \le \varepsilon_{\max}} \ell_U(z + \delta). \tag{10}$$

Using a first-order Taylor expansion around z, we obtain

$$\ell_U(z + \delta) \approx \ell_U(z) + \delta^\top \nabla_z \ell_U(z). \tag{11}$$

The optimal perturbation under the norm constraint is

$$\delta^\star = \varepsilon_{\max} \frac{\nabla_z \ell_U(z)}{\|\nabla_z \ell_U(z)\|}. \tag{12}$$

Substituting $\delta^\star$ into Eq. (11) and taking expectation over the data distribution yields the expected adversarial loss

$$\mathbb{E}\big[\ell_U(z) + \varepsilon_{\max} \|\nabla_z \ell_U(z)\|\big]. \tag{13}$$

Proposition 2 (Implicit gradient regularization). Adversarial self-play is equivalent, to first order, to adding a Jacobian-norm penalty:

$$\mathcal{L}_{U,\text{adv}} = \mathcal{L}_U + \lambda\, \varepsilon_{\max}\, \mathbb{E}\big[\|\nabla_z \ell_U(z)\|\big]. \tag{14}$$

Consequently, the understanding branch is encouraged to reduce its sensitivity to small perturbations in z, leading to locally flatter decision boundaries.

Implication. This explains the empirically observed improvements in robustness: the understanding head learns to be less sensitive to challenging input variations, improving both in-distribution and out-of-distribution performance as well as adversarial robustness.

G.3. Manifold-Expanding Effect of Decoder-Constrained Perturbations

Unlike conventional pixel-space adversarial training, UniGame produces *decoder-constrained* adversarial examples

$$\tilde{x} = G(z + \delta), \quad \tilde{x} \in \mathcal{M}, \tag{15}$$

where G is the decoder and $\mathcal{M}$ is the decodable image manifold. This architecture ensures adversarial samples are:

1. **On-manifold**: $\tilde{x}$ remains realistic and visually plausible;
2. **Semantically valid**: filtered by CLIP-based or similar consistency criteria;
3. **Near boundary regions**: targeted towards regions where the understanding model is fragile.

Assumption 3 (Local bi-Lipschitz decoder). The decoder G is locally bi-Lipschitz on the relevant region of the token space, i.e., there exist constants $0 < m \le M < \infty$ such that for all z_1, z_2 in a neighborhood $\mathcal{Z}$,

$$m\|z_1 - z_2\| \;\le\; \|G(z_1) - G(z_2)\| \;\le\; M\|z_1 - z_2\|. \tag{16}$$

Lemma 1 (Adversarial manifold expansion). Under Assumption 3, for any $z \in \mathcal{Z}$ the support of the perturbed output distribution satisfies

$$\operatorname{supp}(G(z + \mathcal{D})) \supseteq \operatorname{supp}(G(z)), \tag{17}$$

and expands the empirical training distribution toward regions where $\|\nabla_z \ell_U(z)\|$ is large.

Implication. The decoder-constrained perturbations induce a structured "inflation" of the data manifold towards decision boundary regions where the understanding head is uncertain. The hard-sample buffer $\mathcal{B}$ collects such samples, which are approximately located near the understanding decision boundary. Training on $\mathcal{B}$ reduces the empirical risk in these critical regions:

$$\hat{R}_{\text{adv}} = \frac{1}{|\mathcal{B}|} \sum_{x \in \mathcal{B}} \ell_U(x) \tag{18}$$

acts as a surrogate for minimizing the out-of-distribution risk R_{OOD}.

G.4. Summary of Theoretical Insights

The above analysis provides a theoretical lens on the benefits of the UniGame framework:

1. **Convergence of self-play:** Alternating gradient descent–ascent admits a stationary saddle point under mild smoothness and compactness assumptions.
2. **Robust optimization view:** The adversarial creator implicitly enforces a gradient-norm penalty (Eq. (14)), flattening the understanding decision boundary.
3. **Manifold expansion:** Decoder-constrained perturbations generate semantically valid hard samples that expand coverage of the decodable manifold towards challenging regions.
4. **Alignment with empirical gains:** These properties theoretically support the empirical improvements in understanding, consistency, out-of-distribution robustness, and adversarial robustness observed in our experiments.