Temporal Gradient Inversion for Private Trajectory Reconstruction in Embodied Reinforcement Learning

Sudip Bhujel[†] **Shanghao Shi**[‡] **Ruiquan Huang**[†] **Ning Zhang**[‡] **Yang Xiao**[†]
[†]Department of Computer Science, University of Kentucky
[‡]Department of Computer Science and Engineering, Washington University in St. Louis
{sudipbhujel, RuiquanHuang, xiaoy}@uky.edu
{shanghao, zhang.ning}@wustl.edu
https://trace-rl.github.io

Abstract

Distributed learning in embodied reinforcement-learning agents offers a degree of privacy by retaining raw sensor data on-device and transmitting only policy gradients to the server. Yet temporal structure can amplify this leakage beyond single-frame attacks. We introduce **T**emporal **R**econstruction **A**ttack on **C**onsecutive **E**ncodings (TRACE), an amortized temporal gradient-inversion attack that autoregressively reconstructs the sequence of private observation-action trajectories from per-step policy-learning gradients. The attack exploits two structural signals ignored by prior single-frame methods: (i) cross-time correlation between successive embodied gradients, which we formalize via a conditional mutual-information bound, and (ii) closed-form action recovery from policy-head gradient structure, which we prove exact when standard entropy regularization is sufficiently small. On held-out embodied scenes, TRACE reaches 18.8 dB PSNR with near-perfect action recovery at 3–4.5 ms per reconstructed frame, dominating the learning-based baseline across all reconstruction metrics and exceeding optimization attacks while running orders of magnitude faster. Further evaluation demonstrates TRACE's broader applicability across recurrent, residual, and compact transformer victim architectures, multi-modal inputs, and larger discrete action spaces. Defense experiments suggest that protecting temporal gradient streams may require sequence-aware privacy mechanisms.

1 Introduction

Gradient inversion attacks have emerged as a fundamental privacy threat in modern machine learning. By leveraging gradients computed on private input, an adversary can reconstruct the original data with high fidelity [47, 12, 41]. Although initially studied in standard supervised learning settings, this privacy risk becomes particularly relevant wherever gradients are exposed. Notably, in distributed reinforcement learning (RL) scenarios such as robot fleets and shared navigation, the embodied RL agents share model/policy gradients to an external aggregating server while keeping raw environmental observations on-device. In these settings, the privacy risk imposed by gradient inversion attacks becomes aggravated: a recovered embodied trajectory leaks the agent's time-ordered camera view and actions, such as room layout, objects, and where the agent moved, looked, and lingered, enabling activity profiling, premise mapping, and persistent monitoring by a curious server.

From the attacker's perspective, existing gradient inversion techniques target single images or static mini-batches, treating them as i.i.d. samples optimized with simple supervised losses such as cross-entropy [47, 12, 41, 40]. By contrast, embodied policy-learning traces are ordered measurements of a physical process, not i.i.d. samples. Consecutive frames captured by the agent's camera are coupled by transition dynamics: bounded translations, bounded rotations, and a largely persistent 3D

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

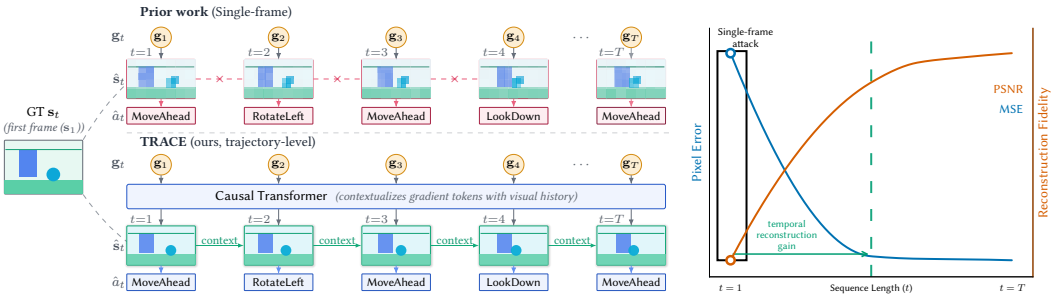


Figure 1: Illustration of trajectory reconstruction from embodied RL gradients. (Left) Prior work inverts each gradient $\mathbf{g}_t$ independently, ignoring temporal coherence; TRACE processes the full trajectory $\{\mathbf{g}_1, \ldots, \mathbf{g}_T\}$ with a causal transformer that sequentially decodes observations ($\hat{\mathbf{s}}_t$) and discrete actions ($\hat{\mathbf{a}}_t$), improving recovery accuracy. (Right) Qualitative illustration of pixel error (MSE) and reconstruction fidelity (PSNR) versus sequence length t: single-frame optimization attacks (t=1) incur high error, while TRACE improves with longer context—realizing temporal reconstruction gain—until returns diminish as $t \to T$ (see Remark D.5).

scene cause nearby observations to share layout, objects, lighting, and viewpoint-dependent geometry. In actor-critic methods such as PPO [30] and A2C [25, 19], the communicated gradients further combine policy, value, and entropy terms whose weights depend on rollout-derived advantages and returns. RL gradient leakage is therefore a trajectory-inversion problem rather than a collection of independent image inversions: the attacker must recover both private RGB observations and discrete actions from a noisy, loss-weighted gradient stream, while the same temporal coherence provides side information that can reduce per-step reconstruction ambiguity. Thus, a valid reconstruction should satisfy the environment's transition dynamics, and those dynamics also provide the structure TRACE exploits.

This work. We introduce TRACE (**T**emporal **R**econstruction **A**ttack on **C**onsecutive **E**ncodings), a temporal gradient inversion attack targeting embodied RL agents through near real-time *inference*. Our key insight is that, in distributed learning settings for embodied agents, consecutive observations leave correlated signatures in their corresponding policy gradients. Unlike prior attacks—whether per-gradient optimization or single-frame learning-based mappers—that invert each gradient independently, TRACE amortizes the attack across the trajectory through an autoregressive model with three components: a gradient encoder maps gradient updates to compact latents, a causal transformer that propagates trajectory context, and a decoder that jointly reconstructs observations and predicts actions. When the server observes per-step gradients from consecutive viewpoints, temporal context amplifies leakage. We substantiate this claim both theoretically, through closed-form discrete-action recovery and an information-theoretic characterization of temporal gain, and empirically. Because policy memory, visual backbones, and auxiliary observations change the gradient representation, we also test whether reconstruction leakage persists beyond the CNN victim.

Our main contributions include:

- We propose TRACE, an amortized trajectory-level inversion framework that uses gradient encoding, causal temporal modeling, and autoregressive decoding to recover observations/actions from gradient streams in milliseconds (Sec. 3).
- We develop a theoretical foundation for actor-critic trajectory-gradient leakage, proving closed-form action recovery and characterizing temporal reconstruction gain (Sec. 4).
- We evaluate TRACE against both optimization-based and learning-based baselines, as well as common gradient defenses, demonstrating high reconstruction fidelity and robust action recovery in realistic embodied-RL settings. The results show temporal gradient leakage transfers across unseen embodied environments, as TRACE achieves strong zero-shot reconstruction on held-out scenes and exhibits rapid improvement when adapted using only a small subset of target-domain trajectories (Sec. 5). To assess the broader applicability of TRACE, we also evaluate recurrent, residual, and transformer victims, multi-modal inputs, and larger discrete action spaces (App. F.3).
- We open-source our implementation of TRACE, which is available at `https://github.com/trace-rl/TRACE`.

2 Problem Formulation

We adopt a standard assumption from gradient inversion research [47, 12, 44, 41, 17]: the model server is *passive* and *honest-but-curious*, faithfully aggregating while observing the client's transmitted gradient updates. We consider the clients are tasked with policy learning via A2C [25] or PPO [30], two widely used actor-critic methods. At each communication round, the server broadcasts a policy model θ_0 to the client. The client then interacts with its private environment using this model, computes per-step gradients from its local rollout, and sends the ordered gradients to the server. Given a contiguous trajectory of gradients $\mathcal{G}_{1:T} = (\mathbf{g}_1, \ldots, \mathbf{g}_T)$ from a victim client where T denotes the trajectory length, the curious server seeks to reconstruct the corresponding trajectory: observations $\hat{\mathbf{s}}_{1:T}$ and actions $\hat{a}_{1:T}$, by minimizing expected reconstruction error and action prediction error.

It is worth noting that cryptographic protections, such as secure aggregation [5], can prevent a server from seeing individual gradients. However, they rely on heavy cryptography methods and introduce extra system/security assumptions that do not apply to the ordered-gradient setting studied here. Under the standard white-box assumption for distributed learning, we assume the adversary (curious server) has full knowledge of model architecture, the current checkpoint θ_0, the action space, and the learning rule, but not the underlying data [47]. The adversary has auxiliary trajectories from the same task family for offline training, after which inference relies solely on the observed ordered gradients.

Formally, let $\mathcal{S} = [0,1]^{3\times H_I \times W_I}$ denote the RGB observation space, with image height H_I and width W_I, and let $\mathcal{A} = \{1, \ldots, K\}$ denote the discrete action space, where $K = |\mathcal{A}|$. A summary of the key symbols used throughout is provided in App. B. For each rollout pair $(\mathbf{s}_t, a_t)$, the client computes a generalized advantage estimate $\hat{A}_t$ and return target $\hat{R}_t$, both derived from the client's local trajectory and hidden from the attacker. We adopt a unified per-step actor-critic objective

$$\mathcal{L}_t^{\mathrm{ac}}(\theta) = \ell_t^{\mathrm{pol}}(\theta) + c_v\big(V_\theta(\mathbf{s}_t) - \hat{R}_t\big)^2 - c_e\,\mathcal{H}(\pi_\theta(\cdot \mid \mathbf{s}_t))\,, \tag{1}$$

where the policy term ℓ_t^{pol} instantiates the underlying algorithm:

$$\ell_t^{\mathrm{pol}}(\theta) \;=\; \begin{cases} -\hat{A}_t \log \pi_\theta(a_t \mid \mathbf{s}_t), & \text{A2C,} \\ -\min\Big(r_t(\theta)\hat{A}_t,\ \mathrm{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\Big), & \text{PPO,} \end{cases} \quad \text{where} \quad r_t(\theta) = \frac{\pi_\theta(a_t \mid \mathbf{s}_t)}{\pi_{\theta_0}(a_t \mid \mathbf{s}_t)}. \tag{2}$$

The attacker observes the corresponding per-step communicated gradient

$$\mathbf{g}_t = \nabla_\theta \mathcal{L}_t^{\mathrm{ac}}(\theta)\big|_{\theta=\theta_0} \in \mathbb{R}^d, \qquad \mathbf{s}_t \in \mathcal{S},\ a_t \in \mathcal{A}. \tag{3}$$

Modeling assumption. We treat $\mathbf{g}_t$ as the per-step gradient at the broadcast parameters θ_0, consistent with per-step policy-gradient sharing in collaborative policy learning [20, 15]. Multi-step client policy updates, defined as $\Delta\theta = \theta_0 - \theta_{\mathrm{client}}$, combine gradients computed along a changing optimization path and may reduce the temporal structure available to the attacker [15]. However, we focus on the per-step setting in this work. Here, $\pi_\theta(\cdot \mid \mathbf{s}_t)$, $V_\theta(\mathbf{s}_t)$, and $\mathcal{H}(\pi_\theta(\cdot \mid \mathbf{s}_t))$ denote the policy, value, and entropy, so the three terms in Equation (1) carry advantage-weighted policy, return-regression, and entropy-regularization signals from both actor and critic branches. A2C [25] and PPO [30] share the same critic and entropy terms and differ only in ℓ_t^{pol}. Later in Proposition 4.1, we show their policy gradients coincide at θ_0, so our attack applies to both without modification.

3 TRACE Design

The core intuition behind TRACE is twofold. First, gradient updates from policy learning tend to reflect the agent's current observation: because the policy gradient $\nabla_\theta \log \pi_\theta(a_t \mid \mathbf{s}_t)$ passes through $\mathbf{s}_t$ via intermediate activations, the gradient vector retains information about the input observation that may be exploited

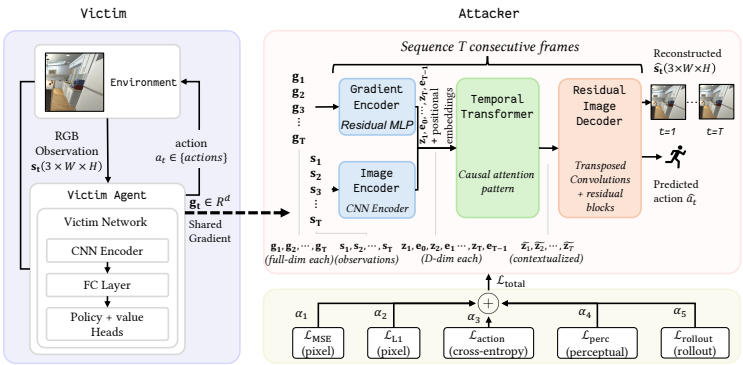


Figure 2: TRACE attack overview.

for reconstruction. Second, embodied RL trajectories exhibit strong temporal coherence—consecutive observations share scene geometry, lighting, and object layout—providing cross-step priors that single-frame inversion methods cannot exploit. As shown in Fig. 2, TRACE leverages both signals jointly: a gradient encoder and an image encoder extract a per-step latent from the gradient updates $\mathbf{g}_{1:T}$; a temporal (causal) transformer propagates scene context across the trajectory. Finally, a residual decoder reconstructs $\hat{\mathbf{s}}_t$ in the pixel space from the contextualized latent and predicts the action $\hat{a}_t$.

3.1 Attacker Model

The attacker's model has three components: i) a gradient encoder that compresses the million-dimensional update into a low-dimensional latent representation; ii) a temporal transformer that contextualizes each latent using causal gradient and image-context tokens; and iii) a decoder that reconstructs the image and predicts the corresponding action.

Encoder. The encoder component comprises a gradient encoder $\phi_{\text{enc}} : \mathbb{R}^d \to \mathbb{R}^D$ and an image encoder $\phi_{\text{img}} : [0,1]^{3\times H_I \times W_I} \to \mathbb{R}^D$, where $D \ll d$ is the shared latent dimension. The gradient encoder maps the high-dimensional raw gradients $\{\mathbf{g}_1, \ldots, \mathbf{g}_T\}$ to compact representations $\mathbf{z}_t = \phi_{\text{enc}}(\mathbf{g}_t) \in \mathbb{R}^D$. The image encoder produces context embeddings. During the teacher-forced warmup: ground-truth observations provide $\mathbf{e}_t = \phi_{\text{img}}(\mathbf{s}_t)$ for $t \geq 1$ while a learned start token $\mathbf{e}_0 \in \mathbb{R}^D$, initialized from $\mathcal{N}(\mathbf{0}, \sigma^2_{\text{init}}\mathbf{I})$, is a shared marker in the parallel sequence. This warmup path forms the interleaved transformer input:

$$\mathbf{X}^{\text{TF}} = [\mathbf{z}_1, \mathbf{e}_0, \mathbf{z}_2, \mathbf{e}_1, \ldots, \mathbf{z}_T, \mathbf{e}_{T-1}] \in \mathbb{R}^{2T\times D}. \tag{4}$$

Because attention is causal, the first reconstruction is conditioned on $\mathbf{z}_1$ and cannot attend to $\mathbf{e}_0$. Similarly, sequential inference begins directly with the first observed gradient.

Temporal Transformer. The causal transformer $\phi_{\text{temp}} : \mathbb{R}^{2T\times D} \to \mathbb{R}^{2T\times D}$ uses N-layer multi-head self-attention with a lower-triangular causal mask $\mathbf{M} \in \{0,1\}^{2T\times 2T}$, $M_{ij} = \mathbf{1}[j \leq i]$. In the teacher-forced warmup sequence of Equation (4), the representation used to decode timestep t is taken from the gradient-token position $\mathbf{z}_t$. Because $\mathbf{e}_{t-1}$ appears immediately after $\mathbf{z}_t$, this parallel warmup path uses lagged image context: $\mathbf{e}_{t-1}$ is available only to later gradient tokens. The sequential scheduled-sampling, rollout, and inference protocol is described in Sec. 3.2: after decoding step t, the encoded ground-truth or predicted frame is appended as context for later steps. Thus, inference is amortized per step but sequential across T decoding steps. The model uses gated residual connections with learnable scalars $\alpha^{(n)}_{\text{attn}}$ and $\alpha^{(n)}_{\text{ffn}}$, initialized so residual branches are initially down-weighted:

$$\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} + \sigma(\alpha^{(n)}_{\text{attn}}) \cdot \text{Attn}^{(n)}(\mathbf{x}^{(n)}) + \sigma(\alpha^{(n)}_{\text{ffn}}) \cdot \text{FFN}^{(n)}(\mathbf{x}^{(n)}). \tag{5}$$

The contextualized gradient representations $\tilde{\mathbf{z}}_t$ are extracted from the output of the transformer.

Decoder. The decoder $\phi_{\text{dec}} : \mathbb{R}^D \to [0,1]^{3\times H_I \times W_I} \times \Delta^{K-1}$ comprises two heads sharing the contextualized latent $\tilde{\mathbf{z}}_t$: an image head $\phi_{\text{img,dec}}$ that upsamples $\tilde{\mathbf{z}}_t$ to a reconstructed observation $\hat{\mathbf{s}}_t$ via transposed convolutions with a sigmoidal output, and an action head ϕ_{act}—a two-layer MLP—that produces a categorical distribution over actions:

$$\hat{\mathbf{s}}_t = \phi_{\text{img,dec}}(\tilde{\mathbf{z}}_t), \qquad \hat{p}_t = \text{softmax}\big(\phi_{\text{act}}(\tilde{\mathbf{z}}_t)\big), \qquad \hat{a}_t = \arg\max_k [\hat{p}_t]_k. \tag{6}$$

3.2 Autoregressive Training

Fig. 3 illustrates the training and inference processes of TRACE. Unlike video-generation settings such as MAGI [46], which generate future frames from prior visual context alone, TRACE conditions on per-step gradients to invert private trajectories. Because TRACE models trajectories sequentially, generating the state at time t depends on the previous predictions. During the initial teacher-forced warmup, the

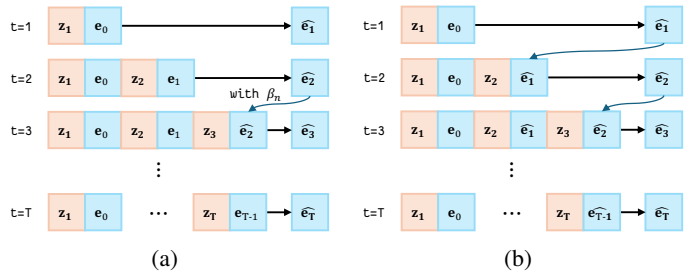


Figure 3: Train–inference protocol of TRACE. (a) During training, after warmup, the next image-context token is encoded from either the ground-truth observation or the model's prediction with probability β_n. (b) During inference, each decoded observation is encoded and appended as image context for subsequent timesteps.

model uses the ground-truth image context in the lagged parallel path of Equation (4). This parallelizes the forward pass and stabilizes early training by preventing compounding errors.

During training, however, teacher forcing introduces *exposure bias* [28]: the model relies on an ideal historical context unavailable during inference. To address this discrepancy, we employ two sequential strategies. First, we employ *scheduled sampling* [4] to gradually replace ground-truth frames with the model's own predictions when constructing future image–context tokens. The epoch-wise replacement probability is

$$\beta_n = \begin{cases} 0, & n \le \tau, \\ \beta_{\text{start}} + (\beta_{\text{end}} - \beta_{\text{start}}) \dfrac{n-\tau}{N-\tau}, & n > \tau, \end{cases} \tag{7}$$

where n denotes the current epoch, τ is the warmup duration, N is the total number of epochs, and $(\beta_{\text{start}}, \beta_{\text{end}})$ are schedule endpoints. Hence, training is fully teacher-forced for $n \le \tau$, then linearly transitions to stronger self-conditioned context usage. Second, we add an auxiliary short-horizon *rollout loss*: from a random start index, we unroll R autoregressive steps, supervise the predictions, where R is the rollout horizon, and feed each predicted frame back as context. Feedback frames are detached to preserve supervision under self-generated context while avoiding full gradient propagation through the feedback path. Inference follows the same sequential feedback pattern over all T timesteps, yielding per-step decoding instead of a parallel full-trajectory pass.

Training Objective. Lastly, we train our attacker model using a weighted multi-term objective, a standard approach in gradient inversion [12, 41, 17, 14], combining complementary loss functions:

$$\mathcal{L}_{\text{total}} = \alpha_1 \mathcal{L}_{\text{MSE}} + \alpha_2 \mathcal{L}_{\text{L1}} + \alpha_3 \mathcal{L}_{\text{act}} + \alpha_4 \mathcal{L}_{\text{perc}} + \alpha_5 \mathcal{L}_{\text{rollout}}. \tag{8}$$

where $\mathcal{L}_{\text{MSE}}$ and $\mathcal{L}_{\text{L1}}$ evaluate pixel fidelity loss, $\mathcal{L}_{\text{act}}$ action supervision loss, $\mathcal{L}_{\text{perc}}$ perceptual quality loss, and $\mathcal{L}_{\text{rollout}}$ rollout-supervision loss that exposes the model to its own predictions during training to mitigate exposure bias. α's are the weights. We provide a full derivation in App. D.5.

4 Theoretical Analysis

We focus on two questions: (i) What information is revealed by the per-step policy gradient? (ii) Why does temporal conditioning enhance trajectory reconstruction? Full proofs are deferred to App. D.

4.1 Gradient Leakage and Action Recoverability

We first establish that at the broadcast checkpoint θ_0, A2C [25] and PPO [30] produce the same communicated gradient, thereby yielding a clean three-term decomposition.

Proposition 4.1 (A2C–PPO gradient equivalence and decomposition)**.** *For each timestep t, there exists a neighborhood $\mathcal{U}_t$ of θ_0 such that* $\text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon) = r_t(\theta)$ *for all $\theta \in \mathcal{U}_t$. Consequently,* $\nabla_\theta \ell_t^{\text{pol,PPO}}(\theta)\big|_{\theta_0} = \nabla_\theta \ell_t^{\text{pol,A2C}}(\theta)\big|_{\theta_0}$*, so both algorithms produce the* same *communicated gradient at θ_0, which decomposes as*

$$\mathbf{g}_t = \underbrace{-\hat{A}_t \nabla_\theta \log \pi_\theta(a_t \mid \mathbf{s}_t)\big|_{\theta_0}}_{\text{policy gradient } \mathbf{g}_t^{(\pi)}} + \underbrace{2c_v \big(V_{\theta_0}(\mathbf{s}_t) - \hat{R}_t\big) \nabla_\theta V_\theta(\mathbf{s}_t)\big|_{\theta_0}}_{\text{value gradient } \mathbf{g}_t^{(v)}} \underbrace{- c_e \nabla_\theta \mathcal{H}\big(\pi_\theta(\cdot \mid \mathbf{s}_t)\big)\big|_{\theta_0}}_{\text{entropy gradient } \mathbf{g}_t^{(e)}}. \tag{9}$$

This decomposition isolates three leakage channels. The policy term carries action information (Theorem 4.2), while value and entropy terms encode additional state-dependent structure.

Next, we show that the action label is recoverable from the policy-head gradient component $\mathbf{g}_t^{(\pi)}$.

Theorem 4.2 (Discrete-Action recovery from the policy-head gradient structure)**.** *Let $\boldsymbol{\ell}_t = \mathbf{W}_a^\top \mathbf{h}(\mathbf{s}_t) \in \mathbb{R}^K$ be the policy logits, $\pi_\theta(\cdot \mid \mathbf{s}_t) = \text{softmax}(\boldsymbol{\ell}_t)$ and $\pi_k := \pi_\theta(k \mid \mathbf{s}_t)$. Assume: **(A1)** the actor and critic heads share no parameters, so V_θ does not depend on $\mathbf{W}_a$; **(A2)** $\hat{A}_t \neq 0$; **(A3)** $\mathbf{h}(\mathbf{s}_t) \in \mathbb{R}^{d_h}_{\ge 0}$ with $\|\mathbf{h}(\mathbf{s}_t)\|_1 > 0$ (e.g., a ReLU activation); **(A4)** the policy is non-degenerate at $\mathbf{s}_t$, $\pi_{a_t} < 1$. Let $\widetilde{\sigma}_k := \sum_{i=1}^{d_h} \big[\nabla_{\mathbf{W}_a} \mathcal{L}_t^{\text{ac}}(\theta)\big]_{i,k}\Big|_{\theta_0}$ denote the column sums of the observed policy-head gradient.*

(i) Exact recovery without entropy regularization. *If $c_e = 0$, then $\widetilde{\sigma}_k = \sigma_k := \hat{A}_t \, \|\mathbf{h}(\mathbf{s}_t)\|_1 \big(\pi_k - \mathbf{1}[k = a_t]\big)$, and the true action is recovered exactly by*

$$a_t = \arg\min_k \widetilde{\sigma}_k \text{ if } \hat{A}_t > 0, \qquad a_t = \arg\max_k \widetilde{\sigma}_k \text{ if } \hat{A}_t < 0. \tag{10}$$

For $K \geq 3$, the sign-agnostic rule $a_t = \arg\max_k |\widetilde{\sigma}_k|$ is equivalent to (10)*; for $K = 2$ the two magnitudes coincide and* (10) *must be used, with* $\mathrm{sign}(\hat{A}_t)$ *inferred from cross-head information such as the value-head gradient (see App. D.2).*

(ii) Robust recovery under entropy regularization. *Let $L_t := \max_k \ell_{t,k} - \min_k \ell_{t,k}$ denote the logit spread, $\pi_{\max} := \max_k \pi_k$, and define the* identifiability gap $\Delta_t := |\hat{A}_t| \, \|\mathbf{h}(\mathbf{s}_t)\|_1 \big(1 - \pi_{a_t} + \min_{k \neq a_t} \pi_k\big) > 0$*, which is strictly positive under (**A4**). If $c_e > 0$ and*

$$2\, c_e\, L_t\, \pi_{\max} \;<\; \frac{\Delta_t}{\|\mathbf{h}(\mathbf{s}_t)\|_1}, \tag{11}$$

then (10) *still recovers a_t exactly.*

Remark 4.3 (Necessity of (**A4**) and deterministic-policy degeneracy)**.** *Assumption (**A4**) is essential: in the one-hot limit $\pi_{a_t} \to 1$, $\pi_k \to 0$ for $k \neq a_t$,* every *column sum vanishes ($\sigma_k \equiv 0$), $\Delta_t = 0$, and* (10) *is ill-posed because the $-\hat{A}_t \log \pi_\theta(a_t \mid \mathbf{s}_t)$ contribution to $\nabla_{\mathbf{W}_a}$ is zero. Thus, the action is unrecoverable from the policy-head gradient block alone for a fully deterministic policy, whereas moderately confident PPO/A2C policies keep Δ_t bounded away from zero.*

Theorem 4.2 gives a closed-form, learning-free recovery rule and shows that it remains exact when entropy regularization is small relative to the identifiability gap. Our default experiments use $K{=}5$ actions, with additional evaluation at $K{=}10$, both in the $K \geq 3$ regime. The near-perfect action recovery in our experiments indicates that the evaluated checkpoints avoid the degenerate regime of Remark 4.3; by Proposition 4.1, the same advantage-weighted policy-gradient structure remains identical for both A2C and PPO at θ_0.

4.2 Temporal Conditioning Benefit

We formalize the non-negativity of temporal gain using a mutual-information analysis.

Definition 4.4 (Temporal-context gain)**.** *The temporal-context gain at step t is the conditional mutual information:*

$$\Delta I_t \;:=\; I(\mathbf{s}_t;\, \mathbf{g}_{1:t-1} \mid \mathbf{g}_t) \;=\; H(\mathbf{s}_t \mid \mathbf{g}_t) - H(\mathbf{s}_t \mid \mathbf{g}_{1:t}) \geq 0, \tag{12}$$

i.e., the residual entropy of $\mathbf{s}_t$, after the current gradient $\mathbf{g}_t$ has been observed, that is resolved by additionally conditioning on the past gradient stream $\mathbf{g}_{1:t-1}$.

We first consider a tractable special case: under a Markov victim policy and one-step (TD(0)) advantage–return estimators with stop-gradient on the bootstrap value, $\mathbf{g}_t$ depends on the trajectory only through $(\mathbf{s}_t, a_t)$ and one-step noise, giving $\mathbf{g}_t \perp\!\!\!\perp \mathbf{g}_{1:t-1} \mid \mathbf{s}_t$ (formalized as Lemma D.3). Generalized Advantage Estimation (GAE) with $\lambda{>}0$ and rollout-level normalization fall outside this case. The resulting scaling, which depends on the specific algorithm, is reported empirically in Sec. 5.

Theorem 4.5 (Temporal reconstruction gain)**.** *Let $(\mathbf{s}_1, \ldots, \mathbf{s}_T)$ be a Markov Decision Process (MDP) trajectory with transition kernel $\mathbf{s}_{t+1} \sim P(\cdot \mid \mathbf{s}_t, a_t)$, and let $\mathrm{MSE}^*(Y) := \min_f \mathbb{E}\big[\|\mathbf{s}_t - f(Y)\|_2^2\big]$ denote the Bayes-optimal mean squared reconstruction error of $\mathbf{s}_t$ from observation Y.*

(i) Non-negativity: *$\Delta I_t \geq 0$ for all t, by non-negativity of conditional mutual information. By the Law of Total Variance, $\mathrm{MSE}^*(\mathbf{g}_{1:t}) \leq \mathrm{MSE}^*(\mathbf{g}_t)$, i.e., temporal conditioning cannot hurt.*

(ii) Exact identity under conditional sufficiency: *under Lemma D.3 (Markov victim policy and causally adapted, un-normalized one-step advantage/return estimators), and condition throughout on θ_0. By Lemma D.3 we have the Markov chain $\mathbf{g}_{1:t-1} \to \mathbf{s}_t \to \mathbf{g}_t$, and therefore*

$$\Delta I_t \;=\; I(\mathbf{s}_t;\, \mathbf{g}_{1:t-1}) \;-\; I(\mathbf{g}_t;\, \mathbf{g}_{1:t-1}) \;\leq\; I(\mathbf{s}_t;\, \mathbf{g}_{1:t-1}) \tag{13}$$

Thus, temporal conditioning helps exactly to the extent that the past gradient stream contains information about $\mathbf{s}_t$ that is not already captured by the current gradient $\mathbf{g}_t$.

Part (ii) is exact only under the conditions of Lemma D.3. Both GAE with $\lambda > 0$ and rollout-level normalization violate these conditions by introducing cross-temporal coupling; the scaling behavior observed under the practical PPO and A2C estimators is therefore discussed informally in Remark D.5 and validated empirically in Sec. 5.

5 Experimental Evaluation

Experimental Setup. We evaluate TRACE against victim policies trained on a point-goal navigation task in the AI2-THOR simulator [18]. Unless otherwise specified, the victim observes RGB frames



Figure 4: Ground-truth and TRACE reconstructions over $T{=}8$ PPO trajectory steps.

downsampled to 84×84 and acts in a five-action discrete space: `MoveAhead` (with a grid step size of 0.25 meters), `RotateLeft` ($15°$), `RotateRight` ($15°$), `LookDown` ($15°$), and `LookUp` ($15°$). The default victim policy is implemented as a shared-CNN actor-critic [19] with separate policy and value heads. It is trained with either PPO [30] or A2C [25], employing GAE advantages as formalized in Sec. 2. Both methods employ the same critic, entropy regularization, and gradient-norm bound. Unless otherwise indicated, we report mean $\pm$ one standard deviation per image. We defer the full definition of evaluation metrics to App. E with additional results to App. F.

5.1 Comparison with Baselines

Fig. 4 shows an example of trajectory inversion result generated by TRACE using $T{=}8$ frames against PPO-based victim, where T is the trajectory length.

We further compare TRACE against DLG [47][1], Inverting Gradients (IG) [12][2], and Learning to Invert (LtI) [40][3]. Because these baselines are designed for supervised classification, we adapt them to our RL setting by replacing the classification loss with a simplified actor-critic surrogate: policy cross-entropy combined with value regression, using fixed coefficients. DLG jointly optimizes a dummy image and soft action distribution via L-BFGS for 300 iterations per step. IG first recovers the discrete action analytically from the policy-head gradient (analogous to iDLG [44]), then optimizes a Gaussian-initialized dummy image with Adam for 4,800 iterations, minimizing gradient cosine similarity with anisotropic TV regularization and clamping to valid pixel bounds. LtI trains an MLP on state-gradient pairs to directly map randomly sub-sampled gradients to reconstructed images and discrete actions. All methods are evaluated on 100 held-out trajectories (with $T{=}8$) for both PPO and A2C victims using consistent metrics.

Table 1: Comparison of TRACE with baselines under PPO and A2C, *Act. Acc.* denotes action accuracy and *Infer.* denotes inference time per frame/image.

	PPO						A2C					
Method	**MSE↓**	**PSNR↑**	**SSIM↑**	**LPIPS↓**	**Act. Acc.↑**	**Infer.**	**MSE↓**	**PSNR↑**	**SSIM↑**	**LPIPS↓**	**Act. Acc.↑**	**Infer.**
DLG [47]	0.289 ± 0.052	5.474 ± 0.826	0.043 ± 0.012	1.224 ± 0.096	16.6	31.0s	0.259 ± 0.072	6.106 ± 1.544	0.036 ± 0.006	1.184 ± 0.127	51.0	76.1s
IG [12]	0.161 ± 0.089	8.727 ± 2.791	0.166 ± 0.087	0.766 ± 0.110	31.5	39.3s	0.114 ± 0.021	9.511 ± 0.810	0.093 ± 0.023	0.686 ± 0.038	32.5	39.6s
LtI [40]	0.023 ± 0.009	16.790 ± 1.708	0.529 ± 0.097	0.671 ± 0.092	**100.0**	1.2ms	0.018 ± 0.006	17.652 ± 1.366	0.508 ± 0.062	0.639 ± 0.071	**100.0**	0.2ms
TRACE	$\mathbf{0.014} \pm 0.007$	$\mathbf{18.771} \pm 1.778$	$\mathbf{0.627} \pm 0.105$	$\mathbf{0.362} \pm 0.070$	**100.0**	4.5ms	$\mathbf{0.014} \pm 0.005$	$\mathbf{18.873} \pm 1.391$	$\mathbf{0.583} \pm 0.061$	$\mathbf{0.358} \pm 0.060$	**100.0**	3.0ms

Table 1 shows that TRACE attains the best performance across all reconstruction metrics under both PPO and A2C victims. Optimization-based baselines perform poorly: under PPO, DLG, and IG, PSNR reaches only 5.47 and 8.73 dB, with action accuracies far below practical usefulness, and the trend persists under A2C (6.11 and 9.51 dB). LtI improves substantially, attaining 16.79 dB (PPO) and 17.65 dB (A2C) PSNR with near-perfect action accuracy, consistent with Theorem 4.2. Adding the causal temporal transformer, TRACE further reduces MSE from 0.023 to 0.014 (PPO) and 0.018 to 0.014 (A2C) over LtI, gains ~ 2 and ~ 1 dB PSNR respectively, and nearly halves LPIPS in both cases (0.671 to 0.362 for PPO, 0.639 to 0.358 for A2C), confirming that exploiting inter-step dependencies yields superior fidelity. We note that TRACE at $T{=}1$ (12.85 dB, Table 4) underperforms LtI's single-frame result, indicating its advantage comes from temporal conditioning rather than a stronger per-frame extractor, consistent with Theorem 4.5. Both learned approaches run in milliseconds per frame, compared with tens of seconds for optimization-based methods, enabling near real-time inversion once the attack model is trained.

[1] https://github.com/mit-han-lab/dlg

[2] https://github.com/JonasGeiping/invertinggradients

[3] https://github.com/wrh14/Learning_to_Invert

We report additional evaluations across victim architectures, multi-modal inputs, and larger action spaces in App. F.3. We also examine four- and eight-step gradient averaging in App. F.4 and auxiliary-data scaling in App. F.5.

5.2 Few-Shot Adaptation

To evaluate cross-scene robustness, we fine-tune the attacker on $p \in \{0, 10, 20, 30, 40, 50\}\%$ of target-domain trajectories, measuring how quickly leakage transfers and how much held-out data is needed for high-fidelity reconstruction. As shown in Table 2, the zero-shot attack provides a strong baseline ($p = 0$) on held-out scenes, with perfect action recovery and reasonable structural fidelity (SSIM $= 0.627$, PSNR $= 18.8$ dB). Even minimal adaptation rapidly improves perception. Fine-tuning with 10% of target trajectories increases PSNR by 2.1 dB and lowers LPIPS to 0.273. With a 50% budget, reconstruction MSE drops to 0.005, and SSIM exceeds 0.74. These results show that TRACE transfers to unseen scenes without target-domain fine-tuning, while a small domain-specific set substantially reduces visual discrepancy, enabling high-fidelity reconstructions.

Table 2: Few-shot adaptation on held-out scenes (PPO). See App. F.2 for A2C results.

p (%)	MSE↓	PSNR↑	SSIM↑	LPIPS↓	Act. Acc.↑
0	$0.014_{\pm 0.007}$	$18.8_{\pm 1.8}$	$0.627_{\pm 0.105}$	$0.362_{\pm 0.070}$	100.0%
10	$0.009_{\pm 0.004}$	$20.9_{\pm 1.8}$	$0.671_{\pm 0.096}$	$0.273_{\pm 0.077}$	100.0%
20	$0.007_{\pm 0.003}$	$22.1_{\pm 2.0}$	$0.697_{\pm 0.095}$	$0.224_{\pm 0.073}$	100.0%
30	$0.006_{\pm 0.003}$	$23.0_{\pm 2.1}$	$0.717_{\pm 0.092}$	$0.186_{\pm 0.063}$	100.0%
40	$0.005_{\pm 0.003}$	$23.4_{\pm 2.2}$	$0.726_{\pm 0.095}$	$0.175_{\pm 0.064}$	100.0%
50	$0.005_{\pm 0.002}$	$23.8_{\pm 2.1}$	$0.740_{\pm 0.092}$	$0.161_{\pm 0.061}$	100.0%

5.3 Defense Evaluation

We evaluate the robustness of TRACE against four gradient defense mechanisms—quantization [3], magnitude-based pruning, additive Gaussian noise, and DP-SGD [1]—under both PPO and A2C victim models, with results shown in Table 3.

Table 3: Evaluation of robustness of defenses under PPO and A2C.

		PPO					A2C				
Defense	**Setting**	**MSE↓**	**PSNR↑**	**SSIM↑**	**LPIPS↓**	**Act. Acc.↑**	**MSE↓**	**PSNR↑**	**SSIM↑**	**LPIPS↓**	**Act. Acc.↑**
No Defense	—	$0.018_{\pm 0.014}$	$18.9_{\pm 3.7}$	$0.628_{\pm 0.143}$	$0.374_{\pm 0.090}$	99.9%	$0.015_{\pm 0.007}$	$18.4_{\pm 1.5}$	$0.530_{\pm 0.068}$	$0.373_{\pm 0.053}$	99.1%
Quantization (bits)	Mild (8-bit)	$0.020_{\pm 0.017}$	$18.5_{\pm 3.9}$	$0.619_{\pm 0.148}$	$0.384_{\pm 0.095}$	99.9%	$0.016_{\pm 0.007}$	$18.4_{\pm 1.5}$	$0.529_{\pm 0.068}$	$0.374_{\pm 0.052}$	98.3%
	4-bit	$0.075_{\pm 0.056}$	$12.4_{\pm 3.2}$	$0.422_{\pm 0.127}$	$0.645_{\pm 0.104}$	37.9%	$0.057_{\pm 0.042}$	$13.3_{\pm 2.7}$	$0.428_{\pm 0.070}$	$0.556_{\pm 0.101}$	50.2%
	2-bit	$0.091_{\pm 0.063}$	$11.3_{\pm 2.8}$	$0.380_{\pm 0.093}$	$0.689_{\pm 0.094}$	19.4%	$0.078_{\pm 0.031}$	$11.5_{\pm 1.8}$	$0.404_{\pm 0.052}$	$0.653_{\pm 0.054}$	41.4%
Pruning (keep %)	90%	$0.018_{\pm 0.014}$	$18.9_{\pm 3.7}$	$0.628_{\pm 0.143}$	$0.374_{\pm 0.090}$	99.9%	$0.015_{\pm 0.007}$	$18.4_{\pm 1.5}$	$0.530_{\pm 0.068}$	$0.373_{\pm 0.053}$	99.1%
	50%	$0.018_{\pm 0.014}$	$18.9_{\pm 3.7}$	$0.628_{\pm 0.143}$	$0.374_{\pm 0.090}$	99.9%	$0.015_{\pm 0.007}$	$18.4_{\pm 1.5}$	$0.530_{\pm 0.068}$	$0.373_{\pm 0.053}$	99.1%
	10%	$0.018_{\pm 0.015}$	$18.8_{\pm 3.8}$	$0.624_{\pm 0.146}$	$0.371_{\pm 0.091}$	99.9%	$0.017_{\pm 0.008}$	$18.0_{\pm 1.7}$	$0.517_{\pm 0.069}$	$0.383_{\pm 0.056}$	89.0%
Noise (σ)	0.001	$0.018_{\pm 0.014}$	$18.8_{\pm 3.8}$	$0.626_{\pm 0.145}$	$0.376_{\pm 0.092}$	99.8%	$0.015_{\pm 0.007}$	$18.4_{\pm 1.5}$	$0.530_{\pm 0.068}$	$0.373_{\pm 0.053}$	98.8%
	0.01	$0.023_{\pm 0.022}$	$18.1_{\pm 4.1}$	$0.599_{\pm 0.160}$	$0.401_{\pm 0.109}$	94.9%	$0.016_{\pm 0.007}$	$18.3_{\pm 1.6}$	$0.525_{\pm 0.069}$	$0.377_{\pm 0.054}$	96.2%
	0.1	$0.049_{\pm 0.034}$	$14.2_{\pm 3.3}$	$0.438_{\pm 0.148}$	$0.572_{\pm 0.111}$	59.9%	$0.026_{\pm 0.016}$	$16.5_{\pm 2.3}$	$0.452_{\pm 0.092}$	$0.440_{\pm 0.081}$	64.1%
DP-SGD (ε, δ)	$(10, 10^{-5})$	$0.069_{\pm 0.037}$	$12.2_{\pm 2.2}$	$0.345_{\pm 0.085}$	$0.658_{\pm 0.085}$	21.4%	$0.050_{\pm 0.018}$	$13.3_{\pm 1.5}$	$0.322_{\pm 0.053}$	$0.563_{\pm 0.068}$	27.4%
	$(5, 10^{-5})$	$0.069_{\pm 0.035}$	$12.1_{\pm 2.1}$	$0.342_{\pm 0.084}$	$0.663_{\pm 0.082}$	21.4%	$0.050_{\pm 0.018}$	$13.2_{\pm 1.4}$	$0.319_{\pm 0.051}$	$0.559_{\pm 0.068}$	23.7%
	$(1, 10^{-5})$	$0.071_{\pm 0.039}$	$12.0_{\pm 2.2}$	$0.346_{\pm 0.084}$	$0.658_{\pm 0.084}$	17.6%	$0.049_{\pm 0.017}$	$13.3_{\pm 1.4}$	$0.322_{\pm 0.053}$	$0.562_{\pm 0.067}$	24.5%

The *No Defense* row provides the clean reference under this defense-evaluation protocol. The attacker is evaluated in a *zero-shot* model, i.e., trained on clean gradients and applied directly to defended ones. The trends are consistent across both algorithms. Pruning is largely ineffective: retaining only 10% of gradient entries leaves PPO reconstruction nearly intact (PSNR $= 18.8$, Act. Acc. $= 99.9\%$), with A2C showing only a modest action accuracy reduction to 89.0%. Quantization to 8-bit is similarly benign, but 4-bit quantization degrades fidelity substantially (PPO PSNR drops to 12.4, A2C to 13.3), and at 2-bit the attack approaches random-chance action recovery for PPO (19.4%). Additive noise impedes reconstruction only at high variance (σ=0.1), where PSNR falls to 14.2 (PPO) and 16.5 (A2C) with action accuracy declining to 60–64%. DP-SGD offers the strongest protection: across all evaluated privacy budgets ($\varepsilon \in \{1, 5, 10\}$), SSIM drops to approximately 0.34 (PPO) and 0.32 (A2C), and action accuracy degrades to near-random levels ($\sim$20%). We do not assess downstream navigation utility under these defenses, as privacy-utility trade-offs require a separate analysis.

5.4 Ablations

We conduct three ablation studies: (i) varying the trajectory length, (ii) restricting the attack to gradients originating from specific architectural components of the victim, and (iii) removing individual terms from the training objective. To isolate the effect of temporal modeling, we employ a

reduced-capacity encoder by halving the hidden dimension from d_{enc}=2048 to d_{enc}=1024. This modification approximately halves the number of encoder parameters while leaving the transformer and decoder components unchanged. For the *Comp.* (gradients from a single victim component) and *Loss (w/o)* (one loss term removed) experiments, the trajectory length is fixed at $T = 8$. The corresponding results are presented in Table 4.

Trajectory Length (T). Extending the horizon from T=1 to T=8 yields a $\sim$ 6 dB PSNR gain (from 12.85 to 18.74) and roughly halves LPIPS (from 0.781 to 0.375). Further extension continues to help modestly: T=64 reaches 20.29 dB PSNR and SSIM 0.666. This trend in sequence length further supports the temporal conditioning mechanism formalized in Theorem 4.5. We adopt T=8 as a balance between temporal context and optimization complexity.[4]

Table 4: Ablation study conducted on the held-out set (PPO). See App. F.7 for A2C results.

	Settings	MSE↓	PSNR↑	SSIM↑	LPIPS↓	Act. Acc.↑
T	1	0.055±0.020	12.85±1.52	0.513±0.078	0.781±0.048	100.0%
	8	0.014±0.006	18.74±1.73	0.617±0.108	0.375±0.071	100.0%
	16	0.015±0.007	18.57±1.78	0.614±0.109	0.395±0.070	100.0%
	32	0.014±0.010	19.62±3.08	0.634±0.131	0.377±0.078	99.8%
	64	0.014±0.012	20.29±4.25	0.666±0.141	0.375±0.091	99.9%
Comp.	CNN	0.015±0.006	18.48±1.72	0.584±0.118	0.384±0.073	100.0%
	FC	0.020±0.013	17.72±2.17	0.605±0.112	0.388±0.074	100.0%
	Heads	0.047±0.029	14.12±2.71	0.530±0.113	0.511±0.088	99.6%
Loss (w/o)	Act.	0.015±0.007	18.56±1.77	0.613±0.108	0.376±0.066	19.0%
	L1	0.018±0.009	17.90±1.88	0.608±0.111	0.409±0.074	100.0%
	LPIPS	0.014±0.006	18.94±1.73	0.627±0.107	0.401±0.088	100.0%
	MSE	0.015±0.007	18.56±1.75	0.612±0.110	0.379±0.070	100.0%

Gradient Components. Restricting the attack to the terminal policy/value *heads* sharply degrades visual fidelity (PSNR = 14.12, LPIPS = 0.511), confirming that high-frequency structural cues reside in the earlier CNN and FC layers. Action recovery, however, stays near-perfect ($\geq$ 99.6%) across all components, indicating that behavioral signals are distributed throughout the network.

Objective Composition. Removing the action loss ('w/o Act.') collapses action accuracy to near-random (19.0%) without improving visual fidelity. Dropping LPIPS marginally raises PSNR to 18.94 but inflates LPIPS to 0.401, while removing L_1 degrades both PSNR (17.90) and LPIPS (0.409). These trade-offs highlight the need for a multi-objective formulation balancing pixel-level distortion and perceptual similarity.

6 Related Work

Learning-based inversion. Classical gradient inversion reconstructs each gradient through iterative optimization [47, 12, 41], incurring per-sample computational overhead that often ranges from minutes to hours. Learning-based attacks amortize this cost with feed-forward inversion models trained on auxiliary data: Wu et al. [40] demonstrated that such models can match or surpass optimization-based methods, even in the presence of gradient compression and perturbation defenses. Subsequent work on efficiency-focused attacks further narrowed this gap [16, 31]. In contrast, TRACE replaces per-gradient optimization entirely with a causal transformer that autoregressively decodes the full trajectory, enabling near real-time inversion without any gradient-by-gradient search.

Gradient inversion attacks in distributed learning. Gradient inversion attacks in distributed or federated learning are well-studied [47, 44, 12, 41, 17, 14, 23]. They may rely on a stronger assumption, wherein the server actively modifies either parameters or model architecture [26, 38, 45, 11]. More recent work extends these attacks to Vision Transformers [14, 23], scales them to larger architectures [31], and systematizes the attack landscape [6]. Unlike these single-sample attacks, TRACE targets temporally correlated gradient streams from embodied policy-learning agents, exploiting cross-step dependencies that prior work treats as i.i.d. noise.

Privacy risks in RL and embodied AI. Gradient inversion tailored to RL and embodied AI remains limited. Li et al. [20] investigated privacy risks for household robots trained via federated RL. At the trajectory level, Du et al. [9] proposed ORL-AUDITOR, an offline deep-RL auditing mechanism that uses cumulative rewards as dataset fingerprints, showing that sequence decision data contains exploitable structure. Recent work also studies federated RL under transition-dynamics constraints [15]. Building on these directions, TRACE exploits the temporal structure across consecutive frames of embodied agents to amortize trajectory-level inversion in near real-time.

[4] Varying T results in different total frame counts and, for T=64 the method transitions to RoPE [32] (see App. F for implementation details). Nevertheless, the observed trend remains consistent for both PPO and A2C (see Table 13).

Defenses. Various defense strategies have also been proposed against gradient inversion, including representation-level protection [33], gradient-leakage resilient FL [37], auditing via generative gradient leakage [21], CENSOR via orthogonal subspace Bayesian sampling [42], SVDefense via singular value decomposition [24], differentially private FL [13], and sparse communication [2]. While these defenses reduce leakage in static settings, their effectiveness against amortized temporal attacks on sequential trajectory data remains open.

7 Conclusion

We introduced TRACE, an amortized temporal gradient inversion attack that reconstructs private observation-action trajectories from ordered policy-learning gradients. Our analysis integrates closed-form discrete-action recovery, an information-theoretic bound on the temporal reconstruction gain, and a local equivalence between A2C and PPO gradients. Across held-out embodied scenes, TRACE achieves 18.8 dB PSNR with near-perfect action recovery in 3–4.5 ms per reconstructed frame, improving over the learning-based baseline by $\sim$2 dB PSNR. Lightweight defenses such as pruning and mild quantization leave substantial leakage, motivating sequence-aware privacy mechanisms and future extensions to continuous-action and off-policy settings, shuffled minibatches, and multi-epoch model updates.

Acknowledgment

This work was supported in part by the US National Science Foundation under grants 2442382 and 2238635, the Office of Naval Research under grants N00014-24-1-2730 and N00014-24-1-2663, and the Army Research Office under grant W911NF-24-1-0155. We also thank the University of Kentucky Center for Computational Sciences and Information Technology Services Research Computing for their support and use of the Lipscomb Compute Cluster (LCC) and associated research computing resources.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[2] Alham Fikri Aji and Kenneth Heafield. Sparse communication for distributed gradient descent. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 440–445, 2017.

[3] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. *Advances in neural information processing systems*, 30, 2017.

[4] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.

[5] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017.

[6] Vincenzo Carletti, Pasquale Foggia, Carlo Mazzocca, Giuseppe Parrella, and Mario Vento. {SoK}: Gradient inversion attacks in federated learning. In *34th USENIX Security Symposium (USENIX Security 25)*, pages 6439–6459, 2025.

[7] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35:16344–16359, 2022.

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*, 2020.

[9] Linkang Du, Min Chen, Mingyang Sun, Shouling Ji, Peng Cheng, Jiming Chen, and Zhikun Zhang. Orl-auditor: Dataset auditing in offline deep reinforcement learning. *arXiv preprint arXiv:2309.03081*, 2023.

[10] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymir Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures, 2018.

[11] Liam Fowl, Jonas Geiping, Wojtek Czaja, Micah Goldblum, and Tom Goldstein. Robbing the fed: Directly obtaining private data in federated learning with modified models. *arXiv preprint arXiv:2110.13057*, 2021.

[12] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients-how easy is it to break privacy in federated learning? *Advances in neural information processing systems*, 33:16937–16947, 2020.

[13] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

[14] Ali Hatamizadeh, Hongxu Yin, Holger R Roth, Wenqi Li, Jan Kautz, Daguang Xu, and Pavlo Molchanov. Gradvit: Gradient inversion of vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10021–10030, 2022.

[15] Shenghong He. Gradient inversion in federated reinforcement learning. *arXiv preprint arXiv:2512.00303*, 2025.

[16] Yangsibo Huang, Samyak Gupta, Zhao Song, Kai Li, and Sanjeev Arora. Evaluating gradient inversion attacks and defenses in federated learning. *Advances in neural information processing systems*, 34:7232–7241, 2021.

[17] Jinwoo Jeon, Kangwook Lee, Sewoong Oh, Jungseul Ok, et al. Gradient inversion with generative image prior. *Advances in neural information processing systems*, 34:29898–29908, 2021.

[18] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

[19] Vijay Konda and John Tsitsiklis. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

[20] Miao Li, Wenhao Ding, and Ding Zhao. Privacy risks in reinforcement learning for household robots. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5148–5154. IEEE, 2024.

[21] Zhuohang Li, Jiaxin Zhang, Luyang Liu, and Jian Liu. Auditing privacy defenses in federated learning via generative gradient leakage. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10132–10142, 2022.

[22] Weisi Lin, Li Dong, and Ping Xue. Visual distortion gauge based on discrimination of noticeable contrast changes. *IEEE transactions on circuits and systems for video technology*, 15(7):900–909, 2005.

[23] Jiahao Lu, Xi Sheryl Zhang, Tianli Zhao, Xiangyu He, and Jian Cheng. April: Finding the achilles' heel on privacy for vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10051–10060, 2022.

[24] Chenxiang Luo, David KY Yau, and Qun Song. Svdefense: Effective defense against gradient inversion attacks via singular value decomposition. *arXiv preprint arXiv:2510.03319*, 2025.

[25] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PmLR, 2016.

[26] Dario Pasquini, Danilo Francati, and Giuseppe Ateniese. Eluding secure aggregation in federated learning via model inconsistency. In *Proceedings of the 2022 ACM SIGSAC Conference on computer and communications security*, pages 2429–2443, 2022.

[27] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[28] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.

[29] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

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

[31] Shanghao Shi, Ning Wang, Yang Xiao, Chaoyu Zhang, Yi Shi, Y Thomas Hou, and Wenjing Lou. Scale-mia: A scalable model inversion attack against secure federated learning via latent space reconstruction. *arXiv preprint arXiv:2311.05808*, 2023.

[32] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[33] Jingwei Sun, Ang Li, Binghui Wang, Huanrui Yang, Hai Li, and Yiran Chen. Soteria: Provable defense against privacy leakage in federated learning from representation perspective. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9311–9319, 2021.

[34] The HDF Group. Hierarchical Data Format, version 5. URL https://github.com/HDFGroup/hdf5.

[35] Zhou Wang and Alan C Bovik. Mean squared error: Love it or leave it? a new look at signal fidelity measures. *IEEE signal processing magazine*, 26(1):98–117, 2009.

[36] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4): 600–612, 2004.

[37] Wenqi Wei, Ling Liu, Yanzhao Wu, Gong Su, and Arun Iyengar. Gradient-leakage resilient federated learning. In *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*, pages 797–807. IEEE, 2021.

[38] Yuxin Wen, Jonas Geiping, Liam Fowl, Micah Goldblum, and Tom Goldstein. Fishing for user data in large-batch federated learning via gradient magnification. *arXiv preprint arXiv:2202.00580*, 2022.

[39] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

[40] Ruihan Wu, Xiangyu Chen, Chuan Guo, and Kilian Q Weinberger. Learning to invert: Simple adaptive attacks for gradient inversion in federated learning. In *Uncertainty in Artificial Intelligence*, pages 2293–2303. PMLR, 2023.

[41] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M Alvarez, Jan Kautz, and Pavlo Molchanov. See through gradients: Image batch recovery via gradinversion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16337–16346, 2021.

[42] Kaiyuan Zhang, Siyuan Cheng, Guangyu Shen, Bruno Ribeiro, Shengwei An, Pin-Yu Chen, Xiangyu Zhang, and Ninghui Li. Censor: Defense against gradient inversion via orthogonal subspace bayesian sampling. *arXiv preprint arXiv:2501.15718*, 2025.

[43] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[44] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. idlg: Improved deep leakage from gradients. *arXiv preprint arXiv:2001.02610*, 2020.

[45] Joshua C Zhao, Atul Sharma, Ahmed Roushdy Elkordy, Yahya H Ezzeldin, Salman Avestimehr, and Saurabh Bagchi. Loki: Large-scale data reconstruction attack against federated learning through model manipulation. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 1287–1305. IEEE, 2024.

[46] Deyu Zhou, Quan Sun, Yuang Peng, Kun Yan, Runpei Dong, Duomin Wang, Zheng Ge, Nan Duan, and Xiangyu Zhang. Taming teacher forcing for masked autoregressive video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7374–7384, 2025.

[47] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. *Advances in neural information processing systems*, 32, 2019.

Appendix

A Broader Impact

In this work, we show that trajectories of private observations and actions can be reconstructed from shared policy gradients in the honest-but-curious setting. Our findings apply to advantage-weighted actor-critic algorithms studied in this paper: the temporal coherence of embodied RL trajectories renders them substantially more vulnerable to gradient inversion than isolated images. Lightweight per-step defenses such as pruning and mild quantization leave substantial cross-step leakage intact. In contrast, stronger perturbations such as DP-SGD, aggressive quantization, and large Gaussian noise degrade reconstruction quality but may also affect utility or communication efficiency. Receiving policy gradients from an embodied agent is thus nearly equivalent to receiving the raw sensor stream, and the amortized attack reconstructs an entire trajectory in milliseconds, making it inexpensive to deploy once trained.

These findings motivate sequence-aware privacy mechanisms that budget protection across an entire rollout rather than treating each gradient update independently. Promising directions include secure aggregation, differential privacy adapted to temporal gradient streams, and defenses that disrupt the cross-step correlations TRACE exploits. However, such mechanisms often incur computational overhead prohibitive for resource-constrained embodied deployments reliant on real-time training, so developing lightweight, sequence-aware defenses that balance privacy and efficiency remains an open challenge. To reduce misuse risk while preserving reproducibility, we do not release pretrained attacker checkpoints or captured trajectory datasets. By exposing the extent of this threat, we hope to accelerate research toward genuinely private collaborative training of embodied RL agents.

B Notations

Table 5 summarizes the key mathematical notations used throughout this paper.

Table 5: Key notations used throughout the paper.

Symbol	Description	Symbol	Description
$t,\ T$	Timestep index and trajectory length	$\theta,\ \theta_0$	Victim parameters and shared checkpoint
$\mathbf{s}_t,\ a_t$	State observation and action at step t	$\hat{A}_t,\ \hat{R}_t$	Advantage and return targets
$\pi_\theta,\ V_\theta$	Policy and value functions	$r_t(\theta)$	PPO importance ratio
$\epsilon,\ c_v,\ c_e$	PPO clip, value, and entropy coefficients	$\mathcal{L}_t^{\text{ac}}$	Per-step actor-critic objective
$\mathbf{g}_t \in \mathbb{R}^d$	Per-step communicated gradient	$\mathcal{G}_{1:T}$	Observed gradient sequence $(\mathbf{g}_1, \dots, \mathbf{g}_T)$
$\phi_{\text{enc}},\ \phi_{\text{img}}$	Gradient and image encoders	$\phi_{\text{temp}},\ \phi_{\text{dec}}$	Temporal transformer and decoder
$\mathbf{z}_t,\ \mathbf{e}_t,\ \tilde{\mathbf{z}}_t$	Gradient token, image-feedback token, and contextualized latent	$D,\ H_I,\ W_I,\ K$	Latent dimension, image size, and action-space size
$\beta_n,\ \tau,\ R$	Scheduled-sampling probability, warmup duration, rollout horizon	$\alpha_1, \dots, \alpha_5$	Training-loss weights
$\mathbf{g}_t^{(\pi)}, \mathbf{g}_t^{(v)}, \mathbf{g}_t^{(e)}$	Policy, value, and entropy gradient components	$\mathbf{W}_a,\ \mathbf{h}(s),\ d_h$	Policy-head weights, hidden representation, hidden width
$\mathbf{G}_a,\ \sigma_k,\ \widetilde{\sigma}_k$	Policy-head gradient, ideal and observed column sums	Δ_t	Identifiability gap
ΔI_t	Temporal information gain	$\text{MSE}^*(\cdot)$	Bayes-optimal reconstruction error
$P(\cdot \mid \mathbf{s}_t, a_t)$	MDP transition kernel		

C Preliminaries

C.1 MDP and Trajectory Notation

We model the victim agent's interaction with the environment as a Markov decision process (MDP) with state space $\mathcal{S}$, action space $\mathcal{A}$, transition kernel $P(\mathbf{s}_{t+1} \mid \mathbf{s}_t, a_t)$, and reward function $r(\mathbf{s}_t, a_t)$. At timestep t, the agent observes state $\mathbf{s}_t$, samples an action $a_t \sim \pi_\theta(\cdot \mid \mathbf{s}_t)$ from a parameterized policy π_θ, receives reward r_t, and transitions to the next state $\mathbf{s}_{t+1}$. A trajectory segment of length T is denoted by

$$\tau_{1:T} = (\mathbf{s}_1, a_1, r_1, \dots, \mathbf{s}_T, a_T, r_T).$$

In our setting, $\mathbf{s}_t$ denotes the private RGB observation at timestep t, and the attacker observes only the communicated gradients derived from such trajectory data.

C.2 Advantage-Weighted Actor-Critic Objective

TRACE targets victims trained with advantage-weighted actor–critic algorithms; we instantiate our analysis and experiments with A2C [25] and PPO [30]. Both learn a stochastic policy $\pi_\theta(a \mid \mathbf{s})$ alongside a critic $V_\theta(\mathbf{s})$, and share the same credit signal, an advantage estimate $\hat{A}_t$ produced by Generalized Advantage Estimation [29],

$$\hat{A}_t = \sum_{k=0}^{\infty} (\gamma\lambda)^k\, \delta_{t+k}, \qquad \delta_t = r_t + \gamma V(\mathbf{s}_{t+1}) - V(\mathbf{s}_t),$$

with reward discount $\gamma \in (0, 1)$, trace-decay $\lambda \in [0, 1]$, and return target $\hat{R}_t = \hat{A}_t + V(\mathbf{s}_t)$. The two algorithms differ only in how the policy is updated.

A2C [25] is a synchronous on-policy actor-critic that follows the advantage-weighted policy gradient

$$\nabla_\theta J^{\text{A2C}}(\theta) = \mathbb{E}_t\big[\nabla_\theta \log \pi_\theta(a_t \mid \mathbf{s}_t)\, \hat{A}_t\big],$$

so the per-step policy loss is the REINFORCE-style [39] estimator

$$\ell_t^{\text{pol,A2C}}(\theta) = -\hat{A}_t\, \log \pi_\theta(a_t \mid \mathbf{s}_t),$$

identical in form to vanilla policy gradient but with $\hat{A}_t$ supplied by the learned critic rather than a Monte-Carlo return. Each rollout is used for one single gradient step and then discarded.

PPO [30] retains the same actor-critic skeleton but replaces the log-likelihood estimator with a clipped importance-weighted surrogate that enables reuse of a rollout over multiple SGD epochs without the updated policy drifting far from the behavior policy. Let θ_0 denote the broadcast policy under which the rollout was collected and define the importance ratio $r_t(\theta) = \pi_\theta(a_t \mid \mathbf{s}_t)/\pi_{\theta_0}(a_t \mid \mathbf{s}_t)$. PPO minimizes

$$\ell_t^{\text{pol,PPO}}(\theta) = -\min\big(r_t(\theta)\hat{A}_t,\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\big),$$

with a small $\epsilon > 0$ capping how far r_t may move before the update is masked.

Let $\ell_t^{\mathrm{pol}}(\theta) \in \{\ell_t^{\mathrm{pol,A2C}}, \ell_t^{\mathrm{pol,PPO}}\}$ denote the algorithm-specific per-step policy loss. Both algorithms then optimize a common per-step objective

$$\mathcal{L}_t^{\mathrm{ac}}(\theta) = \ell_t^{\mathrm{pol}}(\theta) + c_v\big(V_\theta(\mathbf{s}_t) - \hat{R}_t\big)^2 - c_e\,\mathcal{H}\big(\pi_\theta(\cdot \mid \mathbf{s}_t)\big),$$

in which the critic regresses onto $\hat{R}_t$ (weighted by c_v), and the policy is regularized toward higher entropy (weighted by c_e). Because $r_t(\theta_0) = 1$, PPO's clipping is inactive in a neighborhood of θ_0 and the two policy losses produce identical gradients at θ_0; Proposition 4.1 formalizes this equivalence. Consequently, the local gradient decomposition and discrete-action recovery results apply uniformly to both algorithms, while the temporal-gain analysis later imposes additional assumptions on how the advantage and return estimators depend on the trajectory.

C.3 Autoregressive Modeling

An autoregressive model factorizes a joint distribution over a sequence into a product of one-step conditional distributions:

$$p(x_{1:T}) = \prod_{t=1}^{T} p(x_t \mid x_{<t}), \tag{14}$$

where $x_{<t} := (x_1, \ldots, x_{t-1})$. Thus, the prediction at time t is made using only past context.

In TRACE, the target sequence consists of per-timestep reconstructions $(\hat{\mathbf{s}}_t, \hat{a}_t)$, and autoregression arises through the visual context tokens fed into the causal transformer. Let $\mathbf{z}_t = \phi_{\mathrm{enc}}(\mathbf{g}_t)$ denote the encoded gradient at time t, and let $\mathbf{e}_t = \phi_{\mathrm{img}}(\mathbf{s}_t)$ for $t \geq 1$ denote a training-time encoded image-context token. The token $\mathbf{e}_0$ is a learned start token (a trainable parameter, not derived from any image). During the parallel teacher-forced warmup path, the transformer input is interleaved as

$$\mathbf{X} = [\mathbf{z}_1, \mathbf{e}_0, \mathbf{z}_2, \mathbf{e}_1, \ldots, \mathbf{z}_T, \mathbf{e}_{T-1}],$$

so that the prediction at timestep t is conditioned on the current gradient representation $\mathbf{z}_t$, together with the preceding gradient and lagged image-context tokens.

The lower-triangular causal mask $M_{ij} = \mathbf{1}[j \leq i]$ enforces the autoregressive constraint in the parallel path: the gradient token $\mathbf{z}_t$ at position $2t-1$ attends to $\{\mathbf{z}_1, \mathbf{e}_0, \ldots, \mathbf{z}_{t-1}, \mathbf{e}_{t-2}, \mathbf{z}_t\}$, receiving image context up to $\mathbf{e}_{t-2}$. The paired token $\mathbf{e}_{t-1}$ at position $2t$ is visible only to subsequent tokens $\mathbf{z}_{t+1}, \ldots$. Scheduled sampling and inference instead construct the context sequentially: after decoding timestep t, the encoded generated frame (or a sampled ground-truth frame during training) is appended as context for future steps.

D Proof and Derivations

We provide complete proofs of the theoretical results stated in Sec. 4.

D.1 Proof of Proposition 4.1 (A2C–PPO Equivalence and Decomposition)

Proof. We prove the result in three steps: (1) PPO's clipping is locally inactive at θ_0; (2) the gradients of the A2C and PPO policy losses are identical at θ_0; (3) aggregating these terms produces the decomposition.

Step 1 (clipping inactive). By definition of the importance ratio (defined in Sec. 2),

$$r_t(\theta_0) = \frac{\pi_{\theta_0}(a_t \mid \mathbf{s}_t)}{\pi_{\theta_0}(a_t \mid \mathbf{s}_t)} = 1.$$

Since 1 lies in the interior of the clipping interval $[1-\epsilon, 1+\epsilon]$ and $\theta \mapsto r_t(\theta)$ is continuous, there exists a neighborhood $\mathcal{U}_t$ of θ_0 in which $r_t(\theta) \in [1-\epsilon, 1+\epsilon]$ for all $\theta \in \mathcal{U}_t$. Within this neighborhood,

$$\mathrm{clip}\big(r_t(\theta),\, 1-\epsilon,\, 1+\epsilon\big) = r_t(\theta),$$

so the PPO loss reduces to

$$\ell_t^{\mathrm{pol,PPO}}(\theta) = -\hat{A}_t\, r_t(\theta), \qquad \theta \in \mathcal{U}_t. \tag{15}$$

Step 2 (gradient equivalence at θ_0). Writing

$$r_t(\theta) = \exp\Big(\log \pi_\theta(a_t \mid \mathbf{s}_t) - \log \pi_{\theta_0}(a_t \mid \mathbf{s}_t)\Big),$$

where the denominator is constant with respect to θ, and using $r_t(\theta_0) = 1$, the chain rule gives

$$\nabla_\theta r_t(\theta)\big|_{\theta=\theta_0} = r_t(\theta_0) \cdot \nabla_\theta \log \pi_\theta(a_t \mid \mathbf{s}_t)\big|_{\theta=\theta_0} = \nabla_\theta \log \pi_\theta(a_t \mid \mathbf{s}_t)\big|_{\theta=\theta_0}.$$

Therefore

$$\begin{aligned}\nabla_\theta \ell_t^{\mathrm{pol,PPO}}(\theta)\big|_{\theta_0} &= -\hat{A}_t \, \nabla_\theta r_t(\theta)\big|_{\theta_0} = -\hat{A}_t \, \nabla_\theta \log \pi_\theta(a_t \mid \mathbf{s}_t)\big|_{\theta_0} \\ &= \nabla_\theta\big(-\hat{A}_t \log \pi_\theta(a_t \mid \mathbf{s}_t)\big)\big|_{\theta_0} = \nabla_\theta \ell_t^{\mathrm{pol,A2C}}(\theta)\big|_{\theta_0},\end{aligned}$$

establishing the A2C–PPO gradient equivalence at the broadcast checkpoint.

Step 3 (decomposition). Since A2C and PPO share the same critic and entropy terms in Eq. (1) and produce the same policy-term gradient at θ_0 by Step 2, the communicated gradient $\mathbf{g}_t = \nabla_\theta \mathcal{L}_t^{\mathrm{ac}}(\theta)\big|_{\theta_0}$ is identical in both algorithms and can be computed by differentiating either form. Using the A2C policy term for concreteness, we differentiate

$$\mathcal{L}_t^{\mathrm{ac}}(\theta) = -\hat{A}_t \, \log \pi_\theta(a_t \mid \mathbf{s}_t) + c_v\big(V_\theta(\mathbf{s}_t) - \hat{R}_t\big)^2 - c_e \, \mathcal{H}\big(\pi_\theta(\cdot \mid \mathbf{s}_t)\big). \tag{16}$$

Collecting terms:

$$\begin{aligned}\mathbf{g}_t &= \nabla_\theta \mathcal{L}_t^{\mathrm{ac}}(\theta)\big|_{\theta=\theta_0} \\ &= -\hat{A}_t \nabla_\theta \log \pi_\theta(a_t \mid \mathbf{s}_t)\big|_{\theta_0} + 2c_v\big(V_{\theta_0}(\mathbf{s}_t) - \hat{R}_t\big) \, \nabla_\theta V_\theta(\mathbf{s}_t)\big|_{\theta_0} - c_e \, \nabla_\theta \mathcal{H}\big(\pi_\theta(\cdot \mid \mathbf{s}_t)\big)\big|_{\theta_0},\end{aligned}$$

which is Equation (9). □

D.2 Proof of Theorem 4.2 (Discrete-Action Recovery)

We prove the two parts of Theorem 4.2 separately. Part (i) establishes the column-sum identity for the policy contribution; part (ii) bounds the entropy-induced perturbation and gives a sufficient condition under which (10) remains exact.

Proof of part (i): exact recovery when $c_e = 0$.

Proof. Fix a timestep t, and abbreviate $\mathbf{s} = \mathbf{s}_t$, $a = a_t$, and $\hat{A} = \hat{A}_t$. By Proposition 4.1, the policy-term gradient at the broadcast checkpoint θ_0 is

$$\nabla_\theta\big(-\hat{A} \log \pi_\theta(a \mid \mathbf{s})\big)\big|_{\theta_0},$$

for both A2C and PPO. Subsequent reasoning, therefore, applies to both algorithms. Under assumption (A1) the value head shares no parameters with $\mathbf{W}_a$, so $\nabla_{\mathbf{W}_a}(V_\theta(\mathbf{s}) - \hat{R})^2 = \mathbf{0}$. When in addition $c_e = 0$, the entropy term is absent, and the gradient with respect to the policy-head weights $\mathbf{W}_a$ is determined entirely by the policy-loss term.

Let

$$\boldsymbol{\ell} = \mathbf{W}_a^\top \mathbf{h}(\mathbf{s}) \in \mathbb{R}^K, \qquad \boldsymbol{\pi}(\mathbf{s}) = \mathrm{softmax}(\boldsymbol{\ell}),$$

and let $\mathbf{e}_a \in \mathbb{R}^K$ denote the one-hot vector of the true action, i.e. $[\mathbf{e}_a]_k = \mathbf{1}[k = a]$. Since

$$\log \pi(a \mid \mathbf{s}) \;=\; \ell_a \;-\; \log \sum_{j=1}^{K} e^{\ell_j},$$

differentiating with respect to ℓ_k gives the standard softmax identity

$$\frac{\partial \log \pi(a \mid \mathbf{s})}{\partial \ell_k} \;=\; \mathbf{1}[k = a] \;-\; \frac{e^{\ell_k}}{\sum_j e^{\ell_j}} \;=\; \mathbf{1}[k = a] - \pi(k \mid \mathbf{s}), \tag{17}$$

Equivalently, in vector form, $\nabla_{\boldsymbol{\ell}} \log \pi(a \mid \mathbf{s}) = \mathbf{e}_a - \boldsymbol{\pi}(\mathbf{s})$ and consequently

$$\nabla_{\boldsymbol{\ell}}\big(-\hat{A} \log \pi(a \mid \mathbf{s})\big) \;=\; \hat{A}\big(\boldsymbol{\pi}(\mathbf{s}) - \mathbf{e}_a\big).$$

Next, since

$$\ell_k = \sum_{i=1}^{d_h} [\mathbf{W}_a]_{i,k} h_i(\mathbf{s}),$$

we have

$$\frac{\partial \ell_k}{\partial [\mathbf{W}_a]_{i,k'}} \;=\; h_i(\mathbf{s})\, \mathbf{1}[k = k'],$$

Applying the chain rule,

$$\frac{\partial\big(-\hat{A} \log \pi(a \mid \mathbf{s})\big)}{\partial [\mathbf{W}_a]_{i,k}} \;=\; \sum_{k'=1}^{K} \frac{\partial\big(-\hat{A} \log \pi(a \mid \mathbf{s})\big)}{\partial \ell_{k'}} \cdot \frac{\partial \ell_{k'}}{\partial [\mathbf{W}_a]_{i,k}} \;=\; \hat{A}\,\big(\pi(k \mid \mathbf{s}) - \mathbf{1}[k = a]\big)\, h_i(\mathbf{s}). \quad (18)$$

Therefore

$$\mathbf{G}_a \;=\; \frac{\partial\big(-\hat{A} \log \pi(a \mid \mathbf{s})\big)}{\partial \mathbf{W}_a} \;=\; \hat{A} \cdot \mathbf{h}(\mathbf{s})\, \big(\boldsymbol{\pi}(\mathbf{s}) - \mathbf{e}_a\big)^\top, \quad (19)$$

where $\boldsymbol{\pi}(\mathbf{s}) = [\pi(1 \mid \mathbf{s}), \ldots, \pi(K \mid \mathbf{s})]^\top$ and $\mathbf{e}_a$ is the one-hot vector of the true action. Hence

$$[\mathbf{G}_a]_{:,k} = \hat{A} \cdot \mathbf{h}(\mathbf{s})\, \big(\pi(k \mid \mathbf{s}) - \mathbf{1}[k = a]\big). \quad (20)$$

Since $\mathbf{h}(\mathbf{s})$ is the output of a ReLU layer, $h_i(\mathbf{s}) \geq 0$ for all i, and $\|\mathbf{h}(\mathbf{s})\|_1 > 0$ for any non-trivial input. Summing over the hidden dimension:

$$\sigma_k \;:=\; \sum_{i=1}^{d_h} [\mathbf{G}_a]_{i,k} = \hat{A} \cdot \|\mathbf{h}(\mathbf{s})\|_1\, \big(\pi(k \mid \mathbf{s}) - \mathbf{1}[k = a]\big). \quad (21)$$

Under (A4), $\pi(a \mid \mathbf{s}) < 1$, so $1 - \pi(a \mid \mathbf{s}) > 0$ and $\pi(k \mid \mathbf{s}) > 0$ holds for at least one $k \neq a$ (in fact for all k by full softmax support). If $\hat{A} > 0$, then $\sigma_a = -\hat{A}\|\mathbf{h}(\mathbf{s})\|_1(1 - \pi(a \mid \mathbf{s})) < 0$ and $\sigma_k = \hat{A}\|\mathbf{h}(\mathbf{s})\|_1 \pi(k \mid \mathbf{s}) \geq 0$ for $k \neq a$, with equality only if $\pi(k \mid \mathbf{s}) = 0$. Thus $a = \arg\min_k \sigma_k$ (uniquely, since σ_a is strictly the smallest). If $\hat{A} < 0$, the signs reverse and $a = \arg\max_k \sigma_k$. This proves Eq. (10).

For $K \geq 3$, consider the sign-agnostic rule

$$a^* = \arg\max_k |\sigma_k - \bar{\sigma}|, \qquad \bar{\sigma} = \tfrac{1}{K} \textstyle\sum_k \sigma_k,$$

where $\bar{\sigma} = 0$ by $\sum_k \pi(k \mid \mathbf{s}) = 1$. We have $|\sigma_a| = |\hat{A}|\|\mathbf{h}\|_1(1 - \pi(a \mid \mathbf{s}))$ and $|\sigma_k| = |\hat{A}|\|\mathbf{h}\|_1 \pi(k \mid \mathbf{s})$ for $k \neq a$. Because the softmax has full support, $\pi(j \mid \mathbf{s}) > 0$ for every j, so for any $k \neq a$ and $K \geq 3$ we have $\pi(k \mid \mathbf{s}) < 1 - \pi(a \mid \mathbf{s})$ strictly (since $1 - \pi(a \mid \mathbf{s}) = \pi(k \mid \mathbf{s}) + \sum_{j \neq a,k} \pi(j \mid \mathbf{s})$ and the residual sum is positive). Hence $|\sigma_k| < |\sigma_a|$ for all $k \neq a$, and the maximizer is unique at a. For $K = 2$, the two magnitudes coincide, and the sign-aware rule is necessary. □

Proof of part (ii): robust recovery under entropy regularization.

Proof. Reusing the abbreviations $\mathbf{s} = \mathbf{s}_t$, $\hat{A} = \hat{A}_t$, $\boldsymbol{\ell} = \mathbf{W}_a^\top \mathbf{h}(\mathbf{s})$, and $\boldsymbol{\pi} = \mathrm{softmax}(\boldsymbol{\ell})$, we now retain the entropy term $-c_e\, \mathcal{H}(\boldsymbol{\pi})$ in $\mathcal{L}_t^{\mathrm{ac}}$ and quantify its contribution to the column sums.

Step 1 (entropy gradient with respect to logits). Writing the entropy in a convenient form

$$\mathcal{H}(\boldsymbol{\pi}) \;=\; -\sum_{k=1}^{K} \pi_k \log \pi_k \;=\; \log Z - \sum_{k=1}^{K} \pi_k\, \ell_k, \qquad Z := \sum_{j=1}^{K} e^{\ell_j},$$

and using the standard softmax identities $\partial \pi_j / \partial \ell_k = \pi_j(\delta_{jk} - \pi_k)$ and $\partial \log Z / \partial \ell_k = \pi_k$, we obtain

$$\frac{\partial \mathcal{H}}{\partial \ell_k} \;=\; \pi_k - \pi_k\big(\ell_k - \bar{\ell} + 1\big) \;=\; -\,\pi_k\,(\ell_k - \bar{\ell}), \qquad \bar{\ell} := \sum_{j=1}^{K} \pi_j\, \ell_j, \quad (22)$$

or equivalently $\nabla_{\boldsymbol{\ell}} \mathcal{H} = -\,\boldsymbol{\pi} \odot (\boldsymbol{\ell} - \bar{\ell}\,\mathbf{1})$.

Step 2 (entropy gradient with respect to $\mathbf{W}_a$). Composing (22) with $\partial \ell_k / \partial [\mathbf{W}_a]_{i,k'} = h_i(\mathbf{s})\, \mathbf{1}[k = k']$ gives the closed-form expression

$$\nabla_{\mathbf{W}_a} \mathcal{H} \;=\; -\,\mathbf{h}(\mathbf{s}) \left(\boldsymbol{\pi} \odot (\boldsymbol{\ell} - \bar{\ell}\,\mathbf{1})\right)^{\top}.$$

The contribution of the entropy term to column k of the observed gradient is therefore

$$\eta_k \;:=\; -c_e \sum_{i=1}^{d_h} \left[\nabla_{\mathbf{W}_a} \mathcal{H}\right]_{i,k} \;=\; c_e \,\|\mathbf{h}(\mathbf{s})\|_1 \,\pi_k \,(\ell_k - \bar{\ell}), \tag{23}$$

so that the observed column sums decompose as $\widetilde{\sigma}_k = \sigma_k + \eta_k$, where σ_k is the policy-only column sum from part (i).

Step 3 (perturbation bound). Since $|\ell_k - \bar{\ell}| \le L_t$ and $\pi_k \le \pi_{\max}$, (23) gives the uniform bound

$$|\eta_k| \;\le\; c_e \,\|\mathbf{h}(\mathbf{s})\|_1 \,\pi_{\max}\, L_t \qquad \text{for every } k \in \{1, \dots, K\}.$$

Step 4 (invariance of the argmin/argmax). Suppose $\hat{A} > 0$ (the case $\hat{A} < 0$ is symmetric). By part (i), $\sigma_{a_t} < \sigma_k$ for every $k \neq a_t$, with gap

$$\sigma_k - \sigma_{a_t} \;=\; \hat{A}\,\|\mathbf{h}(\mathbf{s})\|_1 \big(1 - \pi_{a_t} + \pi_k\big) \;\ge\; \Delta_t.$$

The rule $\arg\min_k \widetilde{\sigma}_k$ still selects a_t provided $\widetilde{\sigma}_{a_t} < \widetilde{\sigma}_k$ for every $k \neq a_t$, i.e.

$$\eta_k - \eta_{a_t} \;>\; -(\sigma_k - \sigma_{a_t}).$$

A sufficient condition is $|\eta_k - \eta_{a_t}| < \sigma_k - \sigma_{a_t}$ for every $k \neq a_t$, which is in turn implied by the uniform bound $2 \max_k |\eta_k| < \Delta_t$ (since $|\eta_k - \eta_{a_t}| \le 2 \max_k |\eta_k|$ and $\sigma_k - \sigma_{a_t} \ge \Delta_t$). By Step 3, this uniform bound holds whenever $2\, c_e \,\|\mathbf{h}(\mathbf{s})\|_1 \,\pi_{\max}\, L_t < \Delta_t$, i.e. (11). The case $\hat{A} < 0$ follows by reversing every inequality, completing the proof. □

Corollary D.1. *Part (i) of the recovery rule holds regardless of the policy's confidence. Even under a uniform policy $\pi_k = 1/K$ for all k, the true-action column sum satisfies $|\sigma_{a_t}| = |\hat{A}|\,\|\mathbf{h}(\mathbf{s})\|_1\,(1 - 1/K) > 0$, while all other column sums satisfy $|\sigma_k| = |\hat{A}|\,\|\mathbf{h}(\mathbf{s})\|_1/K$. Therefore, for $K \ge 3$ the true action remains the unique maximizer under the sign-agnostic rule (since $(K-1)/K > 1/K$); for $K = 2$ the magnitudes tie, and the sign-aware rule* (10) *is required, with the sign of $\hat{A}_t$ supplied by information outside the policy-head column magnitudes.*

Remark D.2 (Scope of the entropy-safety condition)**.** *Condition* (11) *is a sufficient, not necessary, condition. The standard PPO/A2C entropy coefficients lie in $c_e \in [0.005, 0.05]$, and the right-hand side $\Delta_t / \|\mathbf{h}(\mathbf{s})\|_1 = |\hat{A}_t|\,(1 - \pi_{a_t} + \min_{k \neq a_t} \pi_k)$ remains bounded away from zero whenever $|\hat{A}_t|$ is non-negligible and the policy is not fully deterministic (Remark 4.3). The left-hand side $2 c_e L_t \pi_{\max}$ scales with the logit spread L_t, which in practice is controlled by either (i) standard regularization of the policy head (L_2 weight decay or gradient clipping) or (ii) the entropy bonus itself, which actively penalizes large L_t. Empirically, the near-perfect action recovery reported in Table 1 is consistent with the evaluated checkpoints remaining outside the degenerate regime, but we do not require this sufficient condition to be tight.*

D.3 Conditional Sufficiency of $\mathbf{s}_t$ for $\mathbf{g}_t$ (Lemma D.3)

We first restate the structural lemma that underlies the conditional-sufficiency argument in part (ii) of Theorem 4.5, with the assumptions on the advantage and return estimators stated explicitly. Throughout, we condition only on the broadcast checkpoint θ_0 (no conditioning on realized advantages or returns).

Lemma D.3 (Conditional sufficiency of $\mathbf{s}_t$ for $\mathbf{g}_t$)**.** *Assume:*

(M) Markov victim policy: $a_t \sim \pi_{\theta_0}(\cdot \mid \mathbf{s}_t)$ *depends on the trajectory only through* $\mathbf{s}_t$.

(B) Causally adapted, un-normalized one-step estimators: *there exist deterministic functions $h_A,\ h_R$ such that*

$$\hat{A}_t \;=\; h_A(\mathbf{s}_t, a_t, \xi_t), \qquad \hat{R}_t \;=\; h_R(\mathbf{s}_t, a_t, \xi_t),$$

where $\xi_t := (r_t, \mathbf{s}_{t+1})$ is the one-step Markov noise drawn from the reward kernel $R(\cdot \mid \mathbf{s}_t, a_t)$ and the transition kernel $P(\cdot \mid \mathbf{s}_t, a_t)$.

Then the per-step gradient is the deterministic map

$$\begin{aligned}\mathbf{g}_t \;&=\; G\big(\theta_0, \mathbf{s}_t, a_t, \xi_t\big)\\ &:=\; \nabla_\theta\Big[-h_A(\mathbf{s}_t, a_t, \xi_t)\,\log \pi_\theta(a_t \mid \mathbf{s}_t) + c_v\big(V_\theta(\mathbf{s}_t) - h_R(\mathbf{s}_t, a_t, \xi_t)\big)^2 - c_e\,\mathcal{H}\big(\pi_\theta(\cdot \mid \mathbf{s}_t)\big)\Big]\Big|_{\theta=\theta_0}.\end{aligned}$$

Consequently, conditional on θ_0,

$$\mathbf{g}_t \;\perp\!\!\!\perp\; \big(\{(\mathbf{s}_{t'}, a_{t'})\}_{t'<t},\; \mathbf{g}_{1:t-1}\big) \;\big|\; (\mathbf{s}_t, a_t), \tag{24}$$

and, marginalizing $a_t \sim \pi_{\theta_0}(\cdot \mid \mathbf{s}_t)$,

$$\mathbf{g}_t \;\perp\!\!\!\perp\; \mathbf{g}_{1:t-1} \;\big|\; \mathbf{s}_t. \tag{25}$$

Proof. By Proposition 4.1 the A2C and PPO policy gradients agree at θ_0, so $\mathbf{g}_t$ equals the displayed expression. Substituting the assumed forms of $\hat{A}_t, \hat{R}_t$ from (B) shows that $\mathbf{g}_t = G(\theta_0, \mathbf{s}_t, a_t, \xi_t)$ depends on the trajectory only through $(\mathbf{s}_t, a_t, \xi_t)$. By the Markov property of the MDP, the noise $\xi_t = (r_t, \mathbf{s}_{t+1})$ is, conditional on $(\mathbf{s}_t, a_t)$, drawn from $R(\cdot \mid \mathbf{s}_t, a_t) \otimes P(\cdot \mid \mathbf{s}_t, a_t)$ independently of the past $\{(\mathbf{s}_{t'}, a_{t'})\}_{t'<t}$ and of $\mathbf{g}_{1:t-1}$. Combined with the determinism of G, this yields (24).

To derive (25), write

$$p(\mathbf{g}_t \mid \mathbf{s}_t, \mathbf{g}_{1:t-1}, \theta_0) \;=\; \sum_{a_t \in \mathcal{A}} p(\mathbf{g}_t \mid \mathbf{s}_t, a_t, \mathbf{g}_{1:t-1}, \theta_0)\; \pi_{\theta_0}(a_t \mid \mathbf{s}_t).$$

The first factor equals $p(\mathbf{g}_t \mid \mathbf{s}_t, a_t, \theta_0)$ by (24); the second factor is $\pi_{\theta_0}(a_t \mid \mathbf{s}_t)$ by (M), since a_t is sampled from $\pi_{\theta_0}(\cdot \mid \mathbf{s}_t)$ independently of $\mathbf{g}_{1:t-1}$ given $\mathbf{s}_t$. Both factors are independent of $\mathbf{g}_{1:t-1}$, proving (25). □

Remark D.4 (Scope of assumption (B))**.** *Assumption (B) is satisfied by TD(0) advantage–return pairs computed with stop-gradient on the bootstrap value (e.g. $\hat{A}_t = r_t + \gamma V_{\theta_0}(\mathbf{s}_{t+1}) - V_{\theta_0}(\mathbf{s}_t)$, $\hat{R}_t = r_t + \gamma V_{\theta_0}(\mathbf{s}_{t+1})$). It is* violated *by GAE with $\lambda > 0$, which uses future ξ_{t+k} for $k > 0$, and by rollout-level normalization $\hat{A}_t \leftarrow (\hat{A}_t - \mu)/\sigma$, which couples every timestep through (μ, σ). We discuss the consequences of relaxing (B) in Remark D.5.*

D.4 Proof of Theorem 4.5 (Temporal Reconstruction Gain)

Proof. For part (i), $\Delta I_t = I(\mathbf{s}_t; \mathbf{g}_{1:t-1} \mid \mathbf{g}_t) \geq 0$ by non-negativity of conditional mutual information. For the MSE statement, we work coordinate-wise. Writing $\mathbf{s}_t \in \mathbb{R}^{D_s}$ and using $\mathrm{MSE}^*(Y) = \mathbb{E}[\mathrm{tr}\,\mathrm{Cov}(\mathbf{s}_t \mid Y)] = \sum_{j=1}^{D_s} \mathbb{E}[\mathrm{Var}([\mathbf{s}_t]_j \mid Y)]$, the scalar law of total variance applied to each coordinate gives

$$\mathrm{Var}([\mathbf{s}_t]_j \mid \mathbf{g}_t) \;=\; \mathbb{E}\big[\mathrm{Var}([\mathbf{s}_t]_j \mid \mathbf{g}_{1:t}) \,\big|\, \mathbf{g}_t\big] + \mathrm{Var}\big(\mathbb{E}[[\mathbf{s}_t]_j \mid \mathbf{g}_{1:t}] \,\big|\, \mathbf{g}_t\big) \;\geq\; \mathbb{E}\big[\mathrm{Var}([\mathbf{s}_t]_j \mid \mathbf{g}_{1:t}) \,\big|\, \mathbf{g}_t\big].$$

Taking expectations and summing over j yields $\mathrm{MSE}^*(\mathbf{g}_t) \geq \mathrm{MSE}^*(\mathbf{g}_{1:t})$.

For part (ii), assume (M) and (B) of Lemma D.3 and condition throughout on θ_0. By (25), $\mathbf{g}_t \perp\!\!\!\perp \mathbf{g}_{1:t-1} \mid \mathbf{s}_t$, equivalently $I(\mathbf{g}_t; \mathbf{g}_{1:t-1} \mid \mathbf{s}_t) = 0$. This is exactly the Markov chain

$$\mathbf{g}_{1:t-1} \;\to\; \mathbf{s}_t \;\to\; \mathbf{g}_t.$$

Using the identity $I(X; Y \mid Z) = I(X; Y) - I(Y; Z) + I(Y; Z \mid X)$ with $(X, Y, Z) = (\mathbf{s}_t, \mathbf{g}_{1:t-1}, \mathbf{g}_t)$ and $I(\mathbf{g}_t; \mathbf{g}_{1:t-1} \mid \mathbf{s}_t) = 0$ yields

$$\Delta I_t \;=\; I(\mathbf{s}_t; \mathbf{g}_{1:t-1} \mid \mathbf{g}_t) \;=\; I(\mathbf{s}_t; \mathbf{g}_{1:t-1}) - I(\mathbf{g}_{1:t-1}; \mathbf{g}_t),$$

which is (13). Since mutual information is non-negative, $I(\mathbf{g}_{1:t-1}; \mathbf{g}_t) \geq 0$, and therefore

$$\Delta I_t \;\leq\; I(\mathbf{s}_t; \mathbf{g}_{1:t-1}), \tag{26}$$

proving (26). □

Remark D.5 (Beyond TD(0): GAE and rollout normalization)**.** *The identity* (13) *and bound* (26) *are exact under (M)+(B); the following discussion of the practical PPO/A2C setting is intuition, not a proved correction term. (i)* GAE with λ>0: $\hat{A}_t = \sum_{k\geq 0} (\gamma\lambda)^k\, \delta_{t+k}$ *depends on future noise $\xi_{t:T}$,*

so (B) fails; since the policy gradient at θ_0 scales linearly in $\hat{A}_t$ (Proposition 4.1), the influence of ξ_{t+k} on $\mathbf{g}_t$ is geometrically suppressed by $(\gamma\lambda)^k$ [29]. (ii) Rollout normalization $\hat{A}_t \leftarrow (\hat{A}_t - \mu)/\sigma$ *couples all timesteps through two scalar statistics, breaking the per-step factorization of (B) but only through a low-dimensional channel.*

The ablations in Tables 4 and 13 show different empirical scaling across the two victim algorithms. Both practical settings use GAE and rollout-level advantage normalization, so assumption (B) should be viewed as an idealized reference point rather than an exact description of these runs. A2C plateaus in $[18.3, 18.8]$ dB PSNR for $T \in \{8, 16, 32, 64\}$, whereas PPO continues to improve from 18.74 dB at T=8 to 20.29 dB at T=64. We interpret this as algorithm- and checkpoint-dependent temporal signal strength rather than a theorem-level consequence of the TD(0) idealization.

D.5 Derivation of Training Objective

Following Equation (8), let $\{\mathbf{s}_t\}_{t=1}^T$ denote a ground-truth trajectory and $\{\hat{\mathbf{s}}_t\}_{t=1}^T$ its reconstruction. We begin with two standard pixel-space reconstruction losses, namely the mean-squared error and the ℓ_1 error:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{T}\sum_{t=1}^{T} \|\hat{\mathbf{s}}_t - \mathbf{s}_t\|_2^2, \qquad \mathcal{L}_{L_1} = \frac{1}{T}\sum_{t=1}^{T} \|\hat{\mathbf{s}}_t - \mathbf{s}_t\|_1. \tag{27}$$

To supervise action recovery, we include a cross-entropy loss on the predicted action distribution:

$$\mathcal{L}_{\text{act}} = -\frac{1}{T}\sum_{t=1}^{T} \log \hat{p}_t(a_t), \tag{28}$$

where $\hat{p}_t(a_t)$ denotes the probability assigned to the ground-truth action a_t at timestep t.

Finally, to improve perceptual quality beyond pixel-wise agreement, we further incorporate the LPIPS loss [43]:

$$\mathcal{L}_{\text{perc}} = \frac{1}{T}\sum_{t=1}^{T} \text{LPIPS}(\hat{\mathbf{s}}_t, \mathbf{s}_t). \tag{29}$$

To mitigate exposure bias arising from teacher forcing (see Sec. 3.2), we add a rollout loss $\mathcal{L}_{\text{rollout}}$ that supervises a short autoregressive rollout under self-generated context. Let $\ell(\hat{\mathbf{s}}, \mathbf{s}, \hat{p}, a) = \alpha_1\|\hat{\mathbf{s}} - \mathbf{s}\|_2^2 + \alpha_2\|\hat{\mathbf{s}} - \mathbf{s}\|_1 + \alpha_3(-\log \hat{p}(a)) + \alpha_4 \,\text{LPIPS}(\hat{\mathbf{s}}, \mathbf{s})$ denote the per-step loss. We uniformly sample a start index $t_0 \sim \text{Uniform}\{1, 2, \ldots, T-R+1\}$ and unroll R steps with predicted frames fed back as image context:

$$\mathcal{L}_{\text{rollout}} = \frac{1}{R}\sum_{r=0}^{R-1} \ell(\hat{\mathbf{s}}_{t_0+r},\, \mathbf{s}_{t_0+r},\, \hat{p}_{t_0+r},\, a_{t_0+r}), \tag{30}$$

where $\hat{\mathbf{s}}_{t_0+r} = \phi_{\text{dec}}(\tilde{\mathbf{z}}_{t_0+r})$ uses the previous predicted frame as image context without backpropagating gradients through it. This prevents the model from optimizing the quality of its own context inputs, ensuring the rollout trains robustness to imperfect context rather than learning to produce easily-decoded outputs. We set R=3 in all experiments.

The overall objective combines these terms to jointly enforce reconstruction fidelity, perceptual realism, accurate action prediction, and robustness to autoregressive error accumulation:

$$\mathcal{L}_{\text{total}} = \alpha_1\mathcal{L}_{\text{MSE}} + \alpha_2\mathcal{L}_{\text{L1}} + \alpha_3\mathcal{L}_{\text{act}} + \alpha_4\mathcal{L}_{\text{perc}} + \alpha_5\mathcal{L}_{\text{rollout}}. \tag{31}$$

E Evaluation Metrics

We present the evaluation metrics used to assess the correctness of the reconstructed images.

Mean Squared Error (MSE↓) [35]. It computes the average squared differences between corresponding pixels in the original and reconstructed images. As a pixel-wise distortion metric, lower values indicate better reconstruction.

Peak Signal to Noise Ratio (PSNR↑) [22]. It is a logarithmic transformation of the MSE in decibels, offering an intuitive interpretation where higher values indicate superior image fidelity.

Structural Similarity (SSIM↑) [36]. It evaluates perceptual image quality by comparing luminance, contrast, and structural components between original and reconstructed images, capturing inter-pixel dependencies better than error-based metrics like MSE.

Learned Perceptual Image Patch Similarity (LPIPS↓). [43] It measures perceptual distance using deep features from a pre-trained convolutional neural network and correlates better with human perceptual judgments than traditional metrics. Lower LPIPS values indicate higher perceptual similarity.

Action Accuracy ↑. It reports the percentage of correctly inferred actions from observed gradients, measuring the attacker's ability to recover the victim's navigation decisions. Higher accuracy indicates more severe leakage of private control information.

F Additional Experiment Details

In this section, we provide additional details on the experimental setup and training procedure described in Sec. 5.

Environment and Navigation Task. We use the AI2-THOR simulator [18] with a point-goal navigation task. At the start of each episode, the agent and the target are initialized at randomly selected navigable locations. The objective is to navigate to within a 1.0-meter radius of the target location within a maximum of 200 time steps. A step penalty of -0.01 is applied at each time step, and a sparse terminal reward of $+1.0$ is granted upon successfully reaching the target. To prevent overfitting to specific visual configurations, the training procedures interleave multiple indoor scenes and randomly shuffle the environment at the end of each episode.

Victim Model. The default victim follows a standard actor-critic architecture [19]: a shared convolutional encoder feeds parallel policy and value heads. The encoder is a four-block stack with channel widths $(32, 64, 64, 64)$, kernel sizes $(8, 4, 3, 3)$, and strides $(4, 2, 1, 2)$, and paddings $(2, 1, 1, 1)$, using `ReLU` activations and batch normalization. For an 84×84 RGB input, these settings produce a $5\times5\times64 = 1600$ flattened feature map. The flattened features are passed through a linear projection to a 512-dimensional hidden representation (followed by `ReLU`) that is shared by two linear heads: a policy head over $|\mathcal{A}|=5$ discrete actions and a scalar value head. Including convolution, batch-normalization, fully connected, policy-head, and value-head parameters, the resulting model has a gradient dimension of $d = 936{,}102$, which defines the attacker's input space.

Attacker (TRACE) Model. The attacker model uses the autoregressive architecture described in Sec. 3. The gradient encoder applies a learned input projection $W_{\text{proj}} \in \mathbb{R}^{d\times d_{\text{enc}}}$ followed by residual blocks and a linear map to $\mathbb{R}^D$. The temporal transformer uses $N=6$ layers of pre-normalized multi-head causal self-attention with 8 heads and FlashAttention [7]. By default, the transformer applies learned absolute positional embeddings $\mathbf{P} \in \mathbb{R}^{1\times 2T\times D}$ to the input sequence. For longer sequences ($T=64$), where the fixed embedding table becomes a bottleneck, we switch to Rotary Position Embeddings (RoPE) [32]. The residual decoder upsamples the contextualized latent to 84×84 RGB images and jointly predicts a discrete action distribution. In the standard configuration ($d_{\text{enc}}=2048$, $D=1024$), the model has approximately 2.27 billion parameters, dominated by the input projection ($d \cdot d_{\text{enc}} \approx 1.9 \times 10^9$ weights). In the ablation study, we use $d_{\text{enc}}=1024$, reducing the model to approximately 1.21 billion parameters.

Dataset and Augmentation. We prepare the training dataset by running the victim policy in AI2-THOR [18] and capturing per-step gradients alongside the corresponding RGB observations and actions. The data is stored in `HDF5` [34] format as temporally ordered trajectories. We use 108 out of 120 available indoor scenes for training and reserve the remaining 12 for held-out test scenes. To enhance dataset diversity, we apply color-jitter transformations—specifically perturbations in brightness, contrast, saturation, and hue—to the recorded observations. Subsequently, gradients are recomputed through the frozen victim model using the exact training loss objective, which may be either the PPO surrogate or the A2C policy gradient incorporating GAE advantages, value loss, and entropy.

Computational Requirements. We implement TRACE in Python using PyTorch [27] as the primary framework for both the victim and attacker models. Default attacker training is distributed across six NVIDIA H200 GPUs using PyTorch DistributedDataParallel (DDP); the additional victim experiments use four or six GPUs, as reported in Table 9. These GPUs are not required for the method

Table 6: Hyperparameter settings for TRACE training.

Hyperparameter	**Value**
Sequence length (T)	8
Latent dimension (D)	1024
Encoder hidden dimension (d_{enc})	2048
Transformer layers	6
Attention heads	8
Encoder / Decoder type	residual / residual
Epochs	50
Batch size (per GPU)	8
Gradient accumulation steps	2
Learning rate	1×10^{-4}
Weight decay	1×10^{-5}
LR schedule	cosine (warmup: 5 epochs, min LR: 1×10^{-6})
Scheduled sampling range	$[0.1, 0.8]$ (warmup: 5 epochs)
Rollout steps	3
α_1 (MSE)	1.0
α_2 (L1)	0.5
α_3 (Action Cross Entropy)	0.01
α_4 (LPIPS)	0.2
α_5 (Rollout)	0.5

itself: comparable accelerator clusters such as NVIDIA A100 GPUs are sufficient, with lower-end GPUs mainly increasing training time.

Hyperparameter Settings. Table 6 summarizes the main hyperparameters used for the default TRACE model training configuration. The complete configuration used for experiments is provided in the accompanying `yaml` file with the source code.

F.1 Runtime of Experiments

Table 7 reports end-to-end training runtimes for the main TRACE configurations. The runtimes of the additional victim configurations are reported in Table 9 because these experiments use slightly different, heterogeneous GPU configurations.

F.2 Few-Shot Adaptation

We extend the few-shot adaptation analysis from Sec. 5.2 to the A2C victim policy. Table 8 reports the same adaptation sweep ($p \in \{0, 10, 20, 30, 40, 50\}\%$) on held-out scenes. The trends mirror the PPO setting: zero-shot transfer already achieves near-perfect action recovery and reasonable visual fidelity (PSNR $= 18.9$), while adapting with 50% of target trajectories raises PSNR to 25.0 and lowers LPIPS to 0.132.

Table 8: Few-shot adaptation on held-out scenes (A2C).

p (%)	**MSE**↓	**PSNR**↑	**SSIM**↑	**LPIPS**↓	**Act. Acc.**↑
0	$0.014_{\pm 0.005}$	$18.9_{\pm 1.4}$	$0.583_{\pm 0.061}$	$0.358_{\pm 0.060}$	100.0%
10	$0.009_{\pm 0.007}$	$21.2_{\pm 2.6}$	$0.649_{\pm 0.085}$	$0.252_{\pm 0.087}$	100.0%
20	$0.006_{\pm 0.004}$	$22.7_{\pm 2.4}$	$0.680_{\pm 0.087}$	$0.208_{\pm 0.083}$	100.0%
30	$0.005_{\pm 0.003}$	$23.6_{\pm 2.6}$	$0.703_{\pm 0.092}$	$0.173_{\pm 0.080}$	100.0%
40	$0.004_{\pm 0.002}$	$24.4_{\pm 2.9}$	$0.722_{\pm 0.098}$	$0.148_{\pm 0.080}$	100.0%
50	$0.004_{\pm 0.002}$	$25.0_{\pm 3.1}$	$0.738_{\pm 0.099}$	$0.132_{\pm 0.077}$	100.0%

F.3 Applicability to Victim Configurations

To assess the broader applicability of TRACE, we examine trajectory leakage beyond the default shared-CNN victim with RGB observations and five actions. Alternative visual encoders and a larger residual backbone vary the representation and capacity of the victim, while recurrent state makes current decisions depend on observation history. Multi-modal inputs introduce goal information alongside visual observations. With larger discrete action spaces, we assess whether accurate action recovery persists as the number of candidate actions increases.

Table 7: Runtime of TRACE across various training configurations. *Seq.* indicates the number of gradient sequences used per run. Differences between PPO and A2C arise from the differing sizes of their captured gradient datasets. Reported times, measured in wall-clock hours, encompass training, validation every fifth epoch, and final evaluation on 100 held-out sequences. Unless otherwise specified, reported timings use six NVIDIA H200 GPUs; few-shot adaptation uses two NVIDIA H200 GPUs due to the smaller dataset. These timings reflect our measurement hardware rather than a strict hardware requirement.

Experiment	**Setting**	d_{enc}	D	T	**Epochs**	**PPO**		**A2C**	
						Seq.	**Time (h)**	**Seq.**	**Time (h)**
Base	Full Gradients	2048	1024	8	50	127,188	10.57	149,250	12.92
Sequence length (Tables 4 and 13)	T=1 (single-frame)	1024	1024	1	50	137,013	2.63	162,301	3.43
	T=8	1024	1024	8	50	127,188	8.92	149,250	12.26
	T=16	1024	1024	16	50	115,161	13.79	134,589	18.84
	T=32	1024	1024	32	50	47,541	16.37	54,836	19.15
	T=64 (RoPE)	1024	1024	64	50	12,966	17.62	14,600	20.24
Gradient component (Tables 4 and 13)	CNN layers only	1024	1024	8	50	127,188	6.69	149,250	10.18
	FC layer only	1024	1024	8	50	127,188	8.59	149,250	11.50
	Policy & value heads only	1024	1024	8	50	127,188	6.50	149,250	9.93
Loss ablation (Tables 4 and 13)	w/o MSE	1024	1024	8	50	127,188	8.92	149,250	11.97
	w/o L1	1024	1024	8	50	127,188	9.17	149,250	11.91
	w/o LPIPS	1024	1024	8	50	127,188	8.37	149,250	11.94
	w/o Action	1024	1024	8	50	127,188	8.91	149,250	11.85
Few-shot adaptation (Tables 2 and 8)	10% target data	2048	1024	8	20	531	0.35	1,611	0.54
	20% target data	2048	1024	8	20	1,181	0.44	3,510	0.67
	30% target data	2048	1024	8	20	1,831	0.51	5,431	0.85
	40% target data	2048	1024	8	20	2,359	0.53	7,046	0.99
	50% target data	2048	1024	8	20	3,022	0.61	8,988	1.18

The PPO variants include a compact Vision Transformer (Tiny ViT) [8], two IMPALA-style residual CNNs [10], a CNN with a gated recurrent unit (GRU), a multi-modal CNN, and the original CNN with ten actions. Here, multi-modal inputs comprise RGB observations and an egocentric point-goal vector encoding distance and relative bearing. For each victim, we train a separate TRACE attacker using unaugmented captures rather than the augmented data used to train the base models. The sequences used for each training run are listed in Table 9. All other evaluation settings are kept similar to the base settings. These experiments assess applicability across matched victim settings in AI2-THOR point-goal navigation, rather than the transferability of a single attacker across architectures.

Architectures and observations. Tiny ViT uses 14×14 patches, 128-dimensional embeddings, two transformer layers with four attention heads and MLP width 512, and a class token feeding the policy and value heads. It is a compact ViT implementation, not a pretrained model. The IMPALA-style CNN has three stages with channel widths $(16, 32, 32)$, each comprising a convolution, max pooling, two residual blocks, and group normalization, followed by a 256-unit fully connected layer. The wider variant doubles the channel widths to $(32, 64, 64)$, increasing the victim from 1.09M to 2.37M parameters. CNN+GRU retains the default convolutional stack but uses a 256-unit projection and GRU state. The multi-modal victim fuses the default 512-dimensional visual features with a two-layer, 128-unit goal encoder, then projects to 512 shared features. Its three goal inputs are distance divided by 10 meters and clipped at one, and the sine and cosine of the egocentric goal bearing. The attacker receives gradients only and reconstructs RGB and actions, not the goal vector.

Actions and recurrent capture. The ten-action set adds `MoveBack`, `MoveLeft`, `MoveRight`, `Crouch`, and `Stand` to the default five actions; rotation and look angles remain 15°. For CNN+GRU, hidden state is carried through each episode and reset at episode boundaries. Per-step PPO gradients use the recorded pre-step hidden state as a detached input, without backpropagation through earlier observations. The hidden state is not an attacker input.

Attacker training and cost. As in Table 6, each attacker uses T=8 and trains for 50 epochs with the default transformer, residual decoder, and loss weights. We adapt the default attacker's input dimension and action-head size to match each victim. For the wider IMPALA victim, we also replace

Table 9: Runtime of TRACE for the additional PPO attackers. The full-gradient dimension d equals the number of trainable parameters in the victim model. Reported wall-clock times, in hours, include training, validation every five epochs, and final evaluation on 100 held-out sequences. *Seq.* denotes the number of training sequences, excluding the 5% validation split. Attacker sizes are reported in billions of parameters. All GPUs are NVIDIA H200s.

Victim	K	d	Attacker (B)	Seq.	GPUs	Time (h)
CNN	10	938,667	2.278	46,871	6	5.29
Tiny ViT	5	477,830	1.335	46,770	4	5.46
IMPALA-style CNN	5	1,090,790	2.590	47,118	4	6.85
Wider IMPALA CNN	5	2,373,318	2.180	47,098	6	12.19
CNN+GRU	5	919,462	2.239	47,215	6	7.71
Multi-modal CNN	5	1,281,318	2.980	39,859	6	5.85

Table 10: Additional PPO victims, each with a separately trained attacker and no data augmentation. K is the number of actions.

Victim	K	MSE↓	PSNR (dB)↑	SSIM↑	LPIPS↓	Act. Acc. (%)↑
CNN	10	$0.013_{\pm 0.003}$	$18.92_{\pm 0.96}$	$0.610_{\pm 0.051}$	$0.372_{\pm 0.074}$	99.5
Tiny ViT	5	$0.048_{\pm 0.028}$	$13.99_{\pm 2.81}$	$0.456_{\pm 0.096}$	$0.543_{\pm 0.057}$	99.0
IMPALA-style CNN	5	$0.007_{\pm 0.003}$	$21.85_{\pm 1.93}$	$0.725_{\pm 0.103}$	$0.244_{\pm 0.057}$	97.4
Wider IMPALA CNN	5	$0.007_{\pm 0.005}$	$22.36_{\pm 3.05}$	$0.773_{\pm 0.127}$	$0.208_{\pm 0.106}$	98.5
CNN+GRU	5	$0.015_{\pm 0.006}$	$18.62_{\pm 1.50}$	$0.565_{\pm 0.089}$	$0.390_{\pm 0.072}$	97.5
Multi-modal CNN	5	$0.011_{\pm 0.004}$	$19.78_{\pm 1.64}$	$0.697_{\pm 0.076}$	$0.335_{\pm 0.081}$	97.4

the default single dense input projection with two learned projections, using an intermediate width of 768 and an output width of 2048, to reduce memory use. All runs retain the batch size of 8 per GPU and two accumulation steps in Table 6. Table 9 reports training sequences, model sizes, GPU counts, and elapsed time through the last training/validation record. These times exclude victim training and data generation. Thus, neither compute nor attacker capacity is held fixed across variants, and the offline cost remains substantial despite fast online inference.

Reconstruction results. Table 10 shows 99.5% action accuracy with ten actions. The recurrent and multi-modal victims yield 18.62 and 19.78 dB PSNR, respectively; only RGB and actions are reconstructed in the multi-modal setting. The wider residual victim has 2.37M parameters and yields 22.36 dB PSNR. Tiny ViT retains 99.0% action accuracy but reaches only 13.99 dB PSNR, so reliable action recovery does not imply equally reliable image recovery. Since trajectories, dataset sizes, and attacker capacities differ across victims, differences in reconstruction fidelity cannot be attributed solely to the victim architecture.

F.4 Reconstruction under Gradient Aggregation

TRACE's main evaluation assumes access to ordered per-step gradients. When a learner shares only an aggregate across several steps, the server loses the direct correspondence between individual gradients and observation–action pairs. We therefore examine whether visual content and action proportions can still be reconstructed at the window level, comparing an unadapted per-step attacker, fine-tuning on matched aggregates, and training from scratch on aggregates.

Using the default five-action PPO victim, we average groups of four or eight consecutive stored gradients within each episode, preserve the order of the resulting aggregates, and discard incomplete groups at episode boundaries. Each aggregate is paired with the last RGB frame in its group and the normalized histogram of the actions in that group. The attacker receives $T{=}8$ aggregated gradients per sequence in both settings. Each aggregate corresponds to four original steps in Agg4 or eight in Agg8, so the sequences span 32 and 64 original steps, respectively. This experiment averages existing per-step captures; it does not recompute PPO minibatch updates or shuffle steps across windows.

Training and evaluation. We use an architecture and losses similar to those in Table 6, with action histograms replacing single labels for cross-entropy supervision. Four-step training uses four GPUs and 3 accumulation steps, giving an effective batch size of 96. The from-scratch attacker follows the 50-epoch schedule. For fine-tuning, we initialize from an attacker trained on unaugmented per-step

Table 11: Reconstruction from four- and eight-step gradient averages for the five-action PPO victim. Uniform baselines have no image output.

Attacker	MSE↓	PSNR (dB)↑	SSIM↑	LPIPS↓	Hist. TV↓
Agg4: Four-step averaging					
Before fine-tuning	$0.022_{\pm 0.015}$	$17.90_{\pm 3.80}$	$0.604_{\pm 0.148}$	$0.410_{\pm 0.097}$	$0.576_{\pm 0.226}$
After fine-tuning	$0.021_{\pm 0.014}$	$18.04_{\pm 3.60}$	$0.611_{\pm 0.144}$	$0.412_{\pm 0.097}$	$0.457_{\pm 0.189}$
From scratch	$0.025_{\pm 0.016}$	$17.03_{\pm 3.23}$	$0.578_{\pm 0.140}$	$0.510_{\pm 0.085}$	$0.296_{\pm 0.161}$
Uniform baseline	–	–	–	–	$0.428_{\pm 0.140}$
Agg8: Eight-step averaging					
From scratch	$0.031_{\pm 0.019}$	$16.03_{\pm 3.31}$	$0.568_{\pm 0.142}$	$0.559_{\pm 0.096}$	$0.268_{\pm 0.115}$
Uniform baseline	–	–	–	–	$0.304_{\pm 0.094}$

captures, changing the duration to 20 epochs, learning rate to 10^{-5}, learning-rate warmup to 2 epochs, and minimum learning rate to 10^{-7}. All three four-step evaluations use autoregressive inference on the same first 100 held-out aggregate sequences with stride 8, yielding 800 last-frame targets and action histograms. For eight-step averaging, the from-scratch run uses the same model, losses, 50-epoch schedule, and episode-disjoint splitting procedure. Evaluation uses autoregressive inference on 39 held-out sequences with stride 8, yielding 312 last-frame targets and action histograms. Histogram error is measured by total variation distance, $\mathrm{TV}(\hat{q}, q) = \frac{1}{2}\sum_{k=1}^{5} |\hat{q}_k - q_k|$, where q_k is the observed fraction of action k within the aggregation window and $\hat{q}_k$ is its predicted proportion. Lower TV indicates closer action proportions.

The *uniform baseline* assigns 20% probability to each of five actions in every window, without training or gradients. We divide action counts by the window length, compute the true histogram's TV distance from the uniform prediction, and average over the same evaluation windows as TRACE, separately for each aggregation size. This tests whether TRACE outperforms an equal-frequency guess, because improvement over the unadapted attacker alone does not establish an advantage over this no-gradient reference.

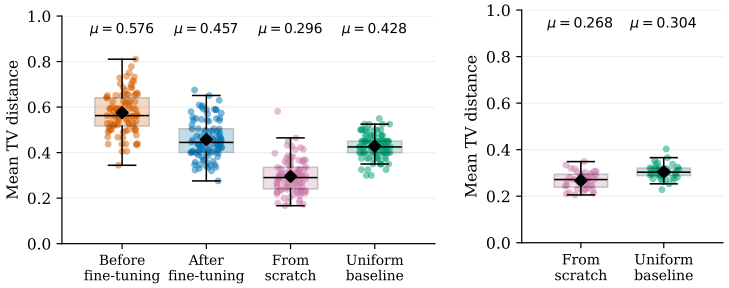
Results. For four-step averaging, Table 11 shows that fine-tuning reduces mean histogram TV from 0.576 to 0.457, while PSNR rises from 17.90 to 18.04 dB and SSIM from 0.604 to 0.611. LPIPS does not improve, increasing slightly from 0.410 to 0.412. Fine-tuning still underperforms the uniform baseline, whose mean TV is 0.428. Training from scratch reduces TV to 0.296, approximately 31% below uniform, with lower sequence-averaged TV in 94 of 100 held-out sequences. Fig. 5 shows these error distributions. Compared with fine-tuning, training from scratch recovers action proportions more accurately but produces less accurate image reconstructions, with PSNR of 17.03 dB and LPIPS of 0.510. The runs use different training budgets and learning rates, so this comparison does not isolate the effect of initialization.

For eight-step averaging, training from scratch yields 16.03 dB PSNR and histogram TV of 0.268, approximately 12% below its uniform baseline of 0.304. Training-set sizes, window targets, and evaluation sample counts differ between aggregation sizes, so these results do not isolate the effect of aggregation size. The image metrics quantify last-frame reconstruction under gradient averaging, not recovery of every observation or its order within a window. These single-run results do not establish robustness to shuffled minibatches, multi-epoch model updates, or secure aggregation.

Overall, training from scratch recovers action proportions more accurately than uniform guessing under both aggregation settings. For Agg4, however, it yields lower PSNR and SSIM and higher MSE and LPIPS than the pretrained attacker, both before and after fine-tuning. The models trained from scratch use 8,736 aggregate training sequences for Agg4 and 2,808 for Agg8. Learning image reconstruction from these smaller datasets without per-step pretraining may contribute to the lower visual fidelity. These experiments do not isolate the effects of dataset size, initialization, and training schedule.

F.5 Effect of Auxiliary Dataset Size

The few-shot study in Sec. 5.2 varies the amount of target-scene data available for adaptation but does not quantify the auxiliary data required to train an attacker. To address this question, we vary the amount of unaugmented auxiliary training data used to train TRACE while holding the victim



(a) Agg4

(b) Agg8

Figure 5: Action-histogram TV for Agg4 and Agg8; lower is better. Reductions in mean TV relative to each panel's uniform baseline indicate improved action-proportion recovery. Each point averages eight windows in one sequence. Boxes show medians and interquartile ranges; diamonds mark means.

Table 12: Auxiliary-data scaling without augmentation for PPO. Percentages select training episodes after a fixed validation holdout; *Seq.* counts training sequences.

Auxiliary data	Seq.	MSE↓	PSNR (dB)↑	SSIM↑	LPIPS↓	Act. Acc. (%)↑
10%	4,846	$0.026_{\pm 0.010}$	$16.25_{\pm 1.80}$	$0.548_{\pm 0.104}$	$0.502_{\pm 0.063}$	100.0
30%	14,267	$0.019_{\pm 0.007}$	$17.66_{\pm 1.78}$	$0.594_{\pm 0.104}$	$0.419_{\pm 0.062}$	100.0
50%	23,694	$0.016_{\pm 0.006}$	$18.24_{\pm 1.65}$	$0.609_{\pm 0.108}$	$0.389_{\pm 0.072}$	100.0
70%	32,981	$0.016_{\pm 0.007}$	$18.44_{\pm 1.78}$	$0.617_{\pm 0.105}$	$0.373_{\pm 0.074}$	100.0
90%	42,329	$0.015_{\pm 0.005}$	$18.60_{\pm 1.67}$	$0.625_{\pm 0.100}$	$0.359_{\pm 0.065}$	100.0
100%	47,313	$0.014_{\pm 0.005}$	$18.71_{\pm 1.62}$	$0.627_{\pm 0.102}$	$0.354_{\pm 0.074}$	100.0

checkpoint and attacker architecture fixed. The dataset comprises 49,800 sequences, of which 2,487 constitute a fixed validation set. We evaluate each attacker on 800 images drawn from the same first 100 sequences of the held-out per-step dataset.

Results. Table 12 reports the completed 10%, 30%, 50%, 70%, 90%, and 100% runs. PSNR increases from 16.25 to 17.66, 18.24, 18.44, 18.60, and 18.71 dB, while LPIPS falls from 0.502 to 0.419, 0.389, 0.373, 0.359, and 0.354. Action accuracy is 100% in all six evaluations. Thus, action recovery remains accurate at the smallest tested fraction, while visual reconstruction benefits from more auxiliary data under this fixed-epoch training protocol. These single-run results do not identify the minimum auxiliary-data requirement or establish robustness to a different task or data domain.

F.6 Qualitative Defense Example

This section complements the defense analysis in Sec. 5.3 with a qualitative example under representative defenses. Fig. 6 shows the reconstructed images for $T{=}8$ under different defense techniques. For this evaluation, we employ the 20% adapted model of the PPO victim to present a visual assessment of the defense's effectiveness.

F.7 Extended Ablation Study

Table 13 repeats the main-body ablations of Table 4 with an A2C victim and makes three observations concrete. *(i) Context window.* Enlarging T from 1 to 8 raises PSNR from 15.54 to 18.60 dB (+3.1 dB), after which A2C plateaus in $[18.26, 18.77]$ dB for $T \in \{16, 32, 64\}$, in contrast to PPO, which continues to improve to 20.29 dB at $T{=}64$ (Table 4). This division is consistent with algorithm-dependent temporal signal strength, while the exact sufficiency result in Lemma D.3 serves as an idealized reference for these practical GAE-based estimators. One possible explanation is that A2C gradients are more strongly tied to the current state, making additional context less informative after short horizons, whereas PPO's use of GAE and rollout-level normalization may preserve cross-timestep dependencies that longer contexts can exploit. *(ii) Gradient components.* CNN-only and FC-only subsets recover PSNR within roughly 0.5 dB of the full gradient, while the policy/value heads alone drop 2.6 dB yet still achieve 100.0% action accuracy—mirroring the PPO pattern in Table 4 and aligning with Theorem 4.2, which predicts exact action recovery from the policy-head structure in our $K{=}5$ setting. *(iii) Loss ablations.* Removing the action loss collapses action accuracy to 16.4% without affecting image PSNR, confirming that the two heads learn complementary signals. Removing LPIPS produces a 0.59 dB PSNR drop for A2C, whereas the PPO ablation (Table 4) shows

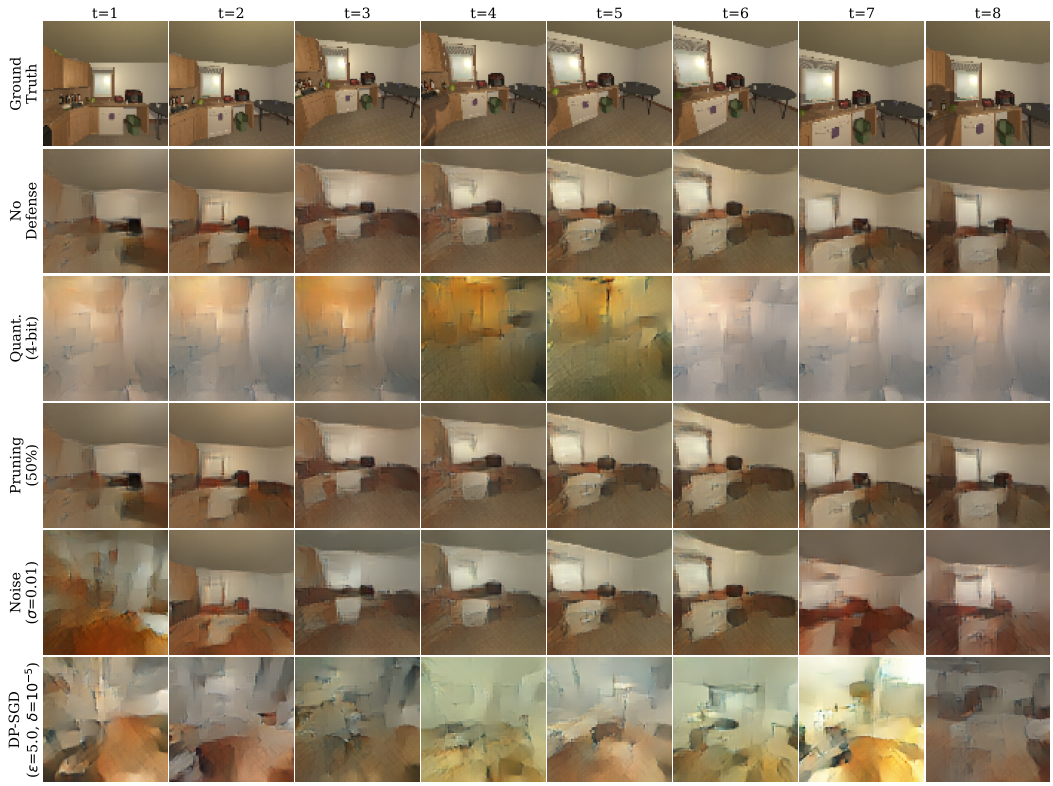


Figure 6: Reconstructions by TRACE under different defense techniques (20%–adapted PPO attacker; corresponding zero-shot results are reported in Table 3).

Table 13: Ablation study conducted on the held-out set (A2C).

	Settings	MSE↓	PSNR↑	SSIM↑	LPIPS↓	Act. Acc.↑
T	1	0.028±0.004	15.54±0.69	0.483±0.034	0.695±0.064	99.0%
	8	0.014±0.004	18.60±1.23	0.564±0.060	0.388±0.064	100.0%
	16	0.015±0.005	18.60±1.31	0.579±0.061	0.375±0.055	98.4%
	32	0.014±0.004	18.77±1.29	0.541±0.068	0.390±0.053	98.9%
	64	0.016±0.007	18.26±1.75	0.560±0.084	0.443±0.061	98.9%
Comp.	CNN	0.016±0.005	18.15±1.20	0.553±0.061	0.391±0.067	99.5%
	FC	0.016±0.007	18.23±1.43	0.558±0.063	0.405±0.069	99.5%
	Heads	0.028±0.013	15.97±1.92	0.500±0.074	0.431±0.091	100.0%
Loss (w/o)	Act.	0.013±0.004	18.88±1.17	0.573±0.058	0.386±0.054	16.4%
	L1	0.014±0.004	18.68±1.23	0.568±0.057	0.388±0.066	100.0%
	LPIPS	0.017±0.007	18.01±1.47	0.567±0.060	0.409±0.067	100.0%
	MSE	0.015±0.004	18.51±1.10	0.564±0.062	0.382±0.064	100.0%

a slight PSNR increase upon removing LPIPS—consistent with the standard perception–distortion tradeoff. We attribute this discrepancy to the different loss landscapes of the two algorithms: the A2C model appears to rely more heavily on the perceptual loss for stable optimization, though the precise mechanism warrants further investigation.

G Discussion

TRACE is designed to reconstruct private trajectories from ordered per-step gradients in near real-time scenarios. Consequently, our evaluation focuses on practical defense mechanisms compatible with these conditions, including pruning, quantization, additive noise, and DP-SGD [1]. DP-SGD reduces reconstruction quality most strongly in our experiments, but its computational costs and privacy–utility

trade-off remain unevaluated. We do not consider heavier defenses such as CENSOR [42] and SVDefense [24], whose overhead are challenging for real-time deployment. App. F.4 evaluates averages of four or eight consecutive stored gradients with last-frame and action-histogram targets, rather than recovery of every observation or its within-window order. These experiments retain ordering across aggregate windows; they do not evaluate shuffled minibatches, recomputed rollout-level updates, multi-epoch client policy updates, or multi-client secure aggregation. These mechanisms combine gradients across time steps, optimization states, or clients, potentially diminishing or eliminating the ordered temporal signal exploited by TRACE.

H Limitations

Our experiments remain confined to point-goal navigation in AI2-THOR [18]. The additional PPO victims cover recurrent, residual, and compact transformer policies, multi-modal inputs, and up to ten discrete actions, but require a separately trained attacker for each victim. They do not establish cross-task or cross-simulator transfer, nor scalability to substantially larger policies. The multi-modal experiment reconstructs RGB and actions, not the goal input; depth, proprioception, and off-policy algorithms remain unevaluated. Recurrent captures condition on detached hidden states rather than backpropagating through complete histories. These settings retain the ordered per-step gradient interface and do not resolve the aggregation limitation above. Finally, our analysis targets discrete-action victims: Theorem 4.2 applies to a categorical policy over a finite set of actions. For Gaussian continuous-control policies, gradients may reveal advantage-weighted residual structure, but identifiability depends on both the covariance parameterization and the advantage-sign information. Consequently, the categorical argmin/argmax rule does not apply directly, and the corresponding identifiability statement is left to future work. TRACE trades offline training on auxiliary trajectories from the same task family for rapid online reconstruction. Our defense study evaluates reconstruction leakage, while downstream navigation utility remains to be assessed.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: *The main claims presented in the abstract and introduction precisely reflect the paper's contributions and scope.*

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: *We provide the limitation of our work in App. H.*

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: *The proofs of all propositions are included in the appendix, and the underlying assumptions are specified in both the main text and the appendix.*

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: *The source code for our implementation is publicly available at* `https://github.com/trace-rl/TRACE`*, along with all configuration details and clear guidelines for reproducing the work.*

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: *The source code and detailed instructions are publicly available at https://github.com/trace-rl/TRACE. The data used in our work can be generated by following the provided code and guidelines. We do not publish the dataset separately, as it is readily reproducible from the provided materials and tailored to our specific experimental setting.*

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: *All details of the experimental setup are described in the main text and the appendix.*

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: *We report mean $\pm$ one standard deviation computed per-image for all reconstruction metrics in the experimental section (Sec. 5). The variability is across individual test images within the held-out evaluation set.*

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: *We provide details on the compute resources in the appendix, particularly in App. F.*

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: *To the best of our knowledge, the research presented in this paper conforms in all respects with the NeurIPS Code of Ethics.*

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: *We describe broader impact in App. A.*

    Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: *Because this is a dual-use privacy attack, we discuss risks and mitigation directions in App. A. We do not release pretrained attacker checkpoints or captured trajectory datasets. The released materials are limited to reproducibility code and instructions.*

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: *We cite and appropriately credit all prior work used in this paper.*

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: *Our source code is released under the MIT license at `https://github.com/trace-rl/TRACE`, along with detailed instructions for use and reproduction.*

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: *Our work does not involve crowdsourcing, experiments with human participants, or research involving human subjects.*

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: *Our work does not involve crowdsourcing, experiments with human participants, or research involving human subjects.*

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: *The core method proposed in the paper does not involve the use of LLMs.*

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.