Feasibility Analysis: Gradient Inversion from Decoded LoRA Adapters

How Difficult Is It Really?
And What Does the "Newer Paper" Actually Do?

Supervisor Notes for Thesis Planning

February 2026

Contents

1 The "Newer Paper" Is NOT What You Think

The paper is **ImpMIA** (Golbari, Wasserman, Vardi & Irani, Weizmann, October 2025): "*Leveraging Implicit Bias for Membership Inference Attack under Realistic Scenarios*." It comes from **your own lab** (Michal Irani + Gal Vardi are co-authors on both papers), but it solves a **fundamentally different problem**.

1.1 Haim et al. vs. ImpMIA: Side-by-Side

	Haim et al. (2022)	**ImpMIA (2025)**
Goal	Reconstruct unknown training images from weights	Identify which *known* candidates were in the training set
Unknowns	Both x (pixels) *and* λ (coefficients)	Only λ — the images are **given**
Task type	Data reconstruction	Membership inference attack
Architecture	MLPs only	ResNet-18
Scale	Dozens of samples	25K training, 50–250K candidate pool
Loss	$\|\nabla\mathcal{L}(x,\lambda) - w\|^2$	$1 - \text{cos\_sim}(A\lambda,\, \theta)$
Classification	Binary	Multiclass (margin-based)

Key Distinction

ImpMIA never reconstructs a single pixel. They take a candidate pool of images, solve the KKT system for λ only (with x fixed), and use the λ coefficients as a membership score. High λ_i means "sample x_i was likely in the training set."

1.2 The ImpMIA Pipeline in Detail

Their method works as follows:

1. **Pre-filter:** For each candidate (x_i, y_i) in the superset X_{sup}, compute the logit margin:
$$\Delta_i = \Phi_{y_i}(\theta;\, x_i) - \max_{j\neq y_i} \Phi_j(\theta;\, x_i) \tag{1}$$
Discard misclassified samples ($\Delta_i < 0$).

2. **Block partition:** Split the model parameters θ into blocks of $\sim$150K parameters each, grouped by layer.

3. **Per-block gradient matrix:** For each block b, compute the per-sample margin gradient:
$$g_i^{(b)} = \nabla_{\theta^{(b)}}\Big[\Phi_{y_i}(\theta;\, x_i) - \max_{j\neq y_i} \Phi_j(\theta;\, x_i)\Big] \tag{2}$$
Stack as columns: $A^{(b)} = [g_1^{(b)} \mid \cdots \mid g_M^{(b)}] \in \mathbb{R}^{p_b \times M}$.

4. **Solve for λ:** For each block, minimize:
$$\mathcal{L} = 1 - \text{cos\_sim}\big(A^{(b)}\lambda^{(b)},\, \theta^{(b)}\big) + \alpha\,\mathcal{L}_{\text{neg}} + \beta\,\mathcal{L}_{\text{marg}} \tag{3}$$
where $\mathcal{L}_{\text{neg}}$ penalizes negative λ entries (KKT complementary slackness) and $\mathcal{L}_{\text{marg}}$ down-weights high-margin samples.

5. **Aggregate:** Collect $\{\lambda_i^{(b)}\}$ across all blocks. Compute trimmed mean and SNR (mean/std across blocks) as robust composite scores.

6. **Post-process:** Apply margin-based boosting and distance scaling. Rank by final score; threshold at desired FPR.

1.3 What IS Useful from ImpMIA for Your Thesis

Despite solving a different problem, three takeaways are genuinely relevant:

Takeaway 1: KKT Conditions Work on ResNet-18

Haim et al. only showed MLPs. ImpMIA shows the implicit bias framework survives batch normalization, skip connections, and non-homogeneity. This is evidence (not proof) that extending to ViTs is plausible.

Takeaway 2: Scaling Engineering

Block-wise parameter partitioning (∼150K per block), cosine similarity loss instead of L2, gradient normalization, and trimmed-mean aggregation. If you apply KKT conditions directly to a ViT (Direction 2), you will need exactly this machinery.

Takeaway 3: Weight Decay Doesn't Break the Theory

Appendix D shows that with weight decay λ_{WD}, the stationarity condition becomes:

$$\theta = \sum_i \ell_i' \cdot \nabla_\theta \Phi(x_i;\theta), \quad \text{where } \ell_i' = -\frac{1}{\lambda_{\text{WD}}}\frac{\partial \ell}{\partial \Phi_i} \tag{4}$$

Same structural form. This matters because every real fine-tuning run uses weight decay.

1.4 What ImpMIA Does NOT Address

× LoRA, PEFT, adapters, or foundation models — zero mention

× Data reconstruction — never recovers pixel content

× Gradient inversion — does not optimize in input space

× Architectures beyond ResNet-18 — no ViTs, no large CNNs

× High-resolution images — only 32×32 (CIFAR)

2 How Difficult Is Gradient Inversion Given Noise?

This is the central risk assessment for the thesis. The answer is: **it depends sharply on the noise level, and nobody has systematically studied this.**

2.1 The Error Pipeline

Your Gradient Bridge pipeline has three stages, each introducing error:

$$\underbrace{BA_{\text{victim}}}_{\text{LoRA adapter}} \xrightarrow[\text{error } \epsilon_1]{\text{Gradient Decoder } f_\phi} \underbrace{\hat{G} \approx \nabla_W \mathcal{L}}_{\text{approx. gradient}} \xrightarrow[\text{error } \epsilon_2]{\text{Gradient Inversion}} \underbrace{\hat{x} \approx x_{\text{train}}}_{\text{reconstructed image}}$$

The total error $\epsilon = \epsilon_1 + \epsilon_2$ compounds. Even if each stage introduces moderate error, the cascade can be catastrophic.

2.2 Stage 1: Decoder Error

R2F reports > 0.9 cosine similarity between decoded and true gradients. This sounds impressive, but consider what it means geometrically.

Cosine Similarity in High Dimensions

For two vectors $\hat{G}, G \in \mathbb{R}^d$ with $\text{cos\_sim}(\hat{G}, G) = c$, we can decompose:

$$\hat{G} = c \frac{\|\hat{G}\|}{\|G\|} G + \sqrt{1 - c^2} \frac{\|\hat{G}\|}{\|G\|} G_\perp \tag{5}$$

where $G_\perp$ is a unit vector orthogonal to G. The **fraction of variance** in the error direction is $1 - c^2$:

cos_sim	Error fraction $(1 - c^2)$
0.99	2%
0.95	9.75%
0.90	19%
0.85	27.75%
0.80	36%

For a ViT-B/16 query projection ($d = 768 \times 768 = 589{,}824$), a cosine similarity of 0.90 means the error component has $0.19 \times 589{,}824 \approx 112{,}000$ effective dimensions of noise. That is enormous.

2.3 Stage 2: Inversion Sensitivity

Gradient inversion (Geiping et al., 2020) solves:

$$\hat{x} = \arg\min_x \left[1 - \text{cos\_sim}\big(\nabla_W \mathcal{L}(W; x),\ G_{\text{target}}\big) + \alpha \cdot \text{TV}(x)\right] \tag{6}$$

where $\text{TV}(x) = \sum_{i,j} |x_{i+1,j} - x_{i,j}| + |x_{i,j+1} - x_{i,j}|$ is the total variation regularizer.

The Fundamental Problem

Equation (6) is a **non-convex** optimization in pixel space. It works well when G_{target} is the **exact** gradient. But as noise is added:

- The loss landscape develops spurious local minima that "explain" the noise
- The optimizer can reconstruct an image that matches the *noisy* gradient but bears no resemblance to the true training image
- The TV regularizer fights noise but also destroys fine detail

2.4 What the Literature Says About Noise Tolerance

Paper	What They Show
Geiping et al. (2020)	Works with exact single-image gradients on ImageNet-scale ResNets. Quality degrades with batch size > 1. **Never tested with approximate gradients.**
Zhu et al. (DLG, 2019)	Extremely sensitive to initialization and noise. Fails on models deeper than a few layers even with exact gradients.
Yin et al. (GradInversion, 2021)	Uses BN statistics as extra signal. More robust, but still assumes exact gradients.
Wei et al. (2020)	Shows differential privacy noise ($\sigma \geq 10^{-3}$) effectively kills gradient inversion.

Critical Gap

Nobody in the gradient inversion literature has systematically studied what happens when gradients are approximate (as opposed to exact + DP noise). This is both a gap and an opportunity — but it means you are walking into unknown territory.

3 Difficulty Assessment by Scenario

Scenario	Difficulty	Feasibility
Perfect gradient, 1 image, small model	Easy	Known to work
Perfect gradient, 1 image, ViT	Medium	Likely works (untested)
Decoded gradient ($\cos\_sim \geq 0.95$), 1 image	Hard	**Unknown — Phase 0 answers this**
Decoded gradient ($\cos\_sim \in [0.85, 0.95)$), 1 image	Very Hard	Likely needs SDS prior
Decoded gradient, batch > 1	Extremely Hard	May not be feasible without strong priors
Multi-step adapter $\rightarrow$ decoded avg. gradient	Extremely Hard	Multiple compounding approximations

4 The Core Tension: Single-Step vs. Multi-Step

This is the issue that should keep you up at night.

R2F trains its decoder on **single-step** LoRA updates. But in a real attack scenario, the victim has trained for T steps. The final adapter is an accumulation:

$$B_T A_T \approx \sum_{t=1}^{T} \eta_t \cdot \nabla_W \mathcal{L}(W_t; x_{b_t}, y_{b_t}) \tag{7}$$

where (x_{b_t}, y_{b_t}) is the mini-batch at step t.

Even if you decode this perfectly into a full-rank gradient, what you recover is an **averaged gradient** — not the gradient at any single training example.

4.1 Why Averaged Gradients Are Harder to Invert

Consider the simplest case: $T = 1$, batch size B. The gradient is:

$$\nabla_W \mathcal{L} = \frac{1}{B} \sum_{i=1}^{B} \nabla_W \ell(W; x_i, y_i) \tag{8}$$

Inverting this to recover all B individual images requires solving:

$$\hat{x}_1, \ldots, \hat{x}_B = \arg\min_{x_1,\ldots,x_B} \left[1 - \text{cos\_sim}\left(\frac{1}{B} \sum_{i=1}^{B} \nabla_W \ell(W; x_i, y_i),\ G_{\text{target}} \right) + \alpha \sum_{i=1}^{B} \text{TV}(x_i) \right] \tag{9}$$

This has $B \times d_{\text{image}}$ unknowns but the constraint comes from a single gradient vector of dimension d_{params}. For $B > 1$, the system is massively underdetermined in the image domain, even though $d_{\text{params}} \gg d_{\text{image}}$.

Compounding Over Training Steps

For T steps with batch size B, the total number of unknown images is potentially $T \times B$, but all information is compressed into a single rank-r adapter. The information bottleneck is severe.

5 Phase 0: The Experiments That Resolve the Uncertainty

Before building the decoder, you must establish the ceiling. Three experiments, in order:

5.1 Experiment 1: The "Cheat" Experiment (Upper Bound)

Setup

1. Fine-tune ViT-B/16 with LoRA on a **single image** for **one step**
2. Record the **exact** full-rank gradient $\nabla_W \mathcal{L}$
3. Feed it into Inverting Gradients (Geiping et al. 2020)
4. Measure: SSIM, PSNR, LPIPS between $\hat{x}$ and x_{train}

Question answered: Can gradient inversion work on ViT at all? If this fails, the entire Gradient Bridge direction is dead regardless of decoder quality.

5.2 Experiment 2: Noise Tolerance Curve (Transfer Function)

Setup

1. Take the exact gradient from Experiment 1
2. Add structured noise at varying levels to achieve target cosine similarities:

$$G_{\text{noisy}} = G + \sigma \cdot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I), \quad \sigma \text{ chosen so } \text{cos\_sim}(G_{\text{noisy}}, G) \in \{0.99, 0.95, 0.90, 0.85, 0.80\} \tag{10}$$

3. Run inversion at each noise level
4. Plot: reconstruction quality (SSIM, PSNR, LPIPS) vs. cosine similarity

Question answered: What cosine similarity does the decoder *need* to achieve for inversion to work? This gives you the spec for the decoder.

5.3 Experiment 3: Batch Size / Averaging Limit

Setup

1. Compute exact gradients for $N \in \{1, 2, 4, 8, 16\}$ images
2. Average them: $\bar{G} = \frac{1}{N}\sum_{i=1}^{N} \nabla_W \ell(W; x_i, y_i)$
3. Attempt to invert $\bar{G}$ to recover individual images
4. Measure: can you recover *any* of the N images? At what quality?

Question answered: What is the practical batch size limit? Beyond what N does reconstruction collapse to a "ghost blend"?

6 Overall Risk Assessment

6.1 Probability of Success by Direction

Direction	P(positive results)	Rationale
Gradient Bridge: single image, single step, on ViT	40–60%	Untested architecture for inversion; decoder noise may be tolerable
Gradient Bridge: realistic (multi-step, batch > 1)	15–25%	Multiple compounding errors; information bottleneck is severe
LoRA in NTK regime (Direction 2)	50–70%	Bypasses decoder entirely; but requires $r \gtrsim N$, limiting practical applicability
SDS generative prior (Direction 3)	60–80%	Diffusion models are powerful priors; but contribution becomes "we added a prior," which is incremental

6.2 The Silver Lining

Even Negative Results Are Publishable

A paper that rigorously characterizes *when* LoRA adapters leak training data and *when they don't* is a perfectly valid contribution. The noise tolerance curve (Experiment 2) would be novel and useful to the privacy community regardless of whether the final reconstruction looks good.

Specifically, a result like: "Gradient inversion from decoded LoRA adapters succeeds for cos_sim ≥ 0.95 but fails catastrophically below 0.90, implying that LoRA rank $r \geq 8$ is necessary for the attack to succeed on ViT-B/16" — that is a publishable finding.

6.3 Recommendation

1. **Do Phase 0 first.** It takes two weeks and tells you everything.
2. **Don't try to make all three directions work simultaneously.** You are spreading too thin.
3. **Let Phase 0 results guide your bet:**
   - If Experiment 1 succeeds $\Rightarrow$ Direction 1 (Gradient Bridge) is viable. Proceed to decoder.
   - If Experiment 1 fails $\Rightarrow$ Pivot to Direction 2 (NTK regime) or Direction 3 (SDS).
   - If Experiment 2 shows graceful degradation $\Rightarrow$ Direction 1 has room for decoder imperfection.
   - If Experiment 2 shows cliff-edge failure $\Rightarrow$ You need the generative prior (Direction 3) no matter what.
4. **Frame the thesis around the characterization**, not just the attack. "Under what conditions do LoRA adapters leak training data?" is a stronger framing than "we reconstruct training data from LoRA."