WHEN SECRETS REVEAL THEMSELVES: VULNERABILITIES OF COVARIANT TRANSFORMATIONS IN PRIVATE LLM INFERENCE AND A LOW-RANK REMEDY

Anonymous authors
Paper under double-blind review

ABSTRACT

Covariant transformations provide an efficient approach to privacy-preserving large language model (LLM) inference by applying secret transformations to private inputs and intermediate representations for privacy protection, while correspondingly transforming the model weights to preserve the original plaintext computation. However, preserving the plaintext computation through gated feed-forward networks (FFNs) and residual connections imposes common structural constraints on existing covariant transformation schemes. With the proposed CROSSWEIGHT, we show that these constraints make it possible to derive an invariant by composing transformed FFN weights, enabling recovery of the secret transformations and then the reconstruction of the private inputs. Across multiple covariant transformation schemes and model architectures, CROSSWEIGHT achieves text-token recovery rates approaching 100%, revealing a critical privacy vulnerability. To mitigate this, we further explore an *irreversible transformation* mechanism based on trainable low-rank transformations, together with an attack-aware objective to guarantee privacy and teacher–student distillation to preserve model utility. Extensive experiments demonstrate that the proposed method substantially reduces the effectiveness of CROSSWEIGHT and existing attacks while achieving a favorable privacy–utility trade-off.

1 INTRODUCTION

Cloud-hosted large language models (LLMs) provide powerful inference capabilities without requiring users to maintain expensive local infrastructure (Sheng et al., 2023; Kwon et al., 2023). However, this centralized paradigm requires users to send prompts to a remote service provider. Such prompts may contain names, addresses, medical records, financial records, or trade secrets, leading to critical privacy risks (Dong et al., 2023; Tong et al., 2025).

Existing private-inference techniques make different trade-offs among security, utility, and efficiency. Cryptographic protocols based on homomorphic encryption (HE), secure multiparty computation (MPC), or hybrid constructions provide rigorous confidentiality guarantees, but incur substantial computation and communication overhead, limiting their practical deployment for large-scale LLM inference (Cheon et al., 2017; Dong et al., 2023; Lu et al., 2025; Moon et al., 2025; Xu et al., 2025). Differentially private mechanisms instead perturb tokens or intermediate representations before transmission (Yue et al., 2021; Du et al., 2023; Mai et al., 2024). While avoiding the excessive computational overhead of cryptographic protocols, such perturbations inevitably distort task-relevant information to achieve privacy, resulting in pronounced utility degradation.

Covariant transformation offers an efficient approach to privacy-preserving LLM inference in terms of both computational efficiency and model utility. Its core idea is to jointly transform private inputs, intermediate representations, and the corresponding model weights using secret transformations, thereby concealing the plaintext inputs and intermediate representations from the inference server while preserving the underlying plaintext computation. As illustrated in Figure 1, for a generic linear mapping with input X and weight W, the server observes only their transformed counterparts, $\widetilde{X} = XP_{\text{in}}$ and $\widetilde{W} = Q_{\text{in}}WP_{\text{out}}$. Here, P_{in}, Q_{in}, and P_{out} are kept secret from the server. Specifically, P_{in} and Q_{in} form paired transformations satisfying $P_{\text{in}}Q_{\text{in}} = I$. This property ensures that

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

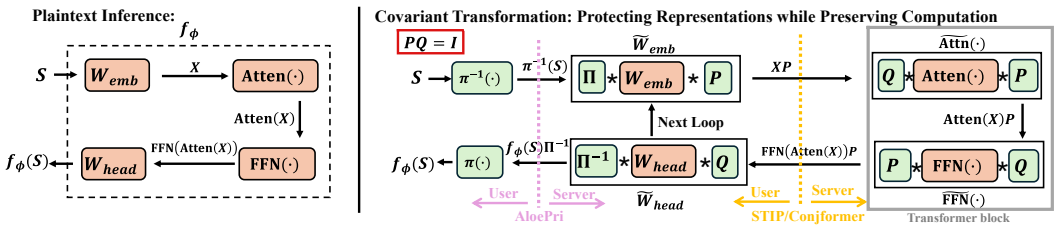


Figure 1: Illustration of covariant transformation-based private LLM inference.

$\widetilde{X}\widetilde{W} = XWP_{\text{out}}$, preserving the original computation up to the output-side transformation P_{out}. Crucially, even when the plaintext weight W is known, the two-sided form $\widetilde{W} = Q_{\text{in}}WP_{\text{out}}$ prevents the server from directly isolating the secret transformations, thereby protecting the privacy of X. By chaining such transformations across adjacent modules, the forward pass can proceed entirely over transformed representations without directly exposing their plaintext counterparts. Representative methods instantiate such transformations differently: STIP (Yuan et al., 2026) primarily employs permutation transformations, CONJFORMER (Yukhimchuk et al., 2026) uses orthogonal transformations, whereas AloePri (Lin et al., 2026) combines multiple transformation structures, including permutations, expansion–contraction transformations, and random scalings, across different model components. Moreover, unlike STIP and CONJFORMER, which transmit transformed representations to the server, AloePri applies a vocabulary-level permutation and transmits permuted token indices, a design intended for practical LLM-as-a-Service (LLMaaS) deployment, as illustrated in Figure 1. More details are provided in the Related Work section in Appendix A. These schemes offer substantially lower overhead than cryptographic approaches and have demonstrated strong empirical robustness against existing input reconstruction attacks (Kugler et al., 2021; Morris et al., 2023; Lin et al., 2026) while maintaining model utility. Particularly, AloePri has been integrated into industrial applications as suggested in Lin et al. (2026).

In this work, we demonstrate that existing covariant transformation schemes remain vulnerable because preserving gated feed-forward network (FFN) and residual computations imposes strong constraints on the admissible transformations. Specifically, nonlinear activations and Hadamard gating restrict FFN intermediate transformations to permutations and diagonal scalings, while residual connections require paired transformations across residual-connected branches. These structural constraints induce exploitable relations among the transformed model weights. To exploit this common structural weakness, we unify STIP, CONJFORMER, and AloePri under a common FFN transformation framework. We show that composing the transformed up and down projection weights in FFN yields an invariant. Under the known-reference-model setting adopted in (Lin et al., 2026), this invariant enables recovery of the secret FFN transformation, which is then used to recover the input transformation and ultimately the private input. We explore this insight in CROSSWEIGHT, a two-stage attack that reconstructs private prompts with text-token recovery rates approaching 100% across the aforementioned covariant transformation schemes with various model architectures, revealing a critical vulnerability in existing covariant transformation-based private inference schemes. To mitigate this vulnerability, we further propose an *irreversible transformation* mechanism based on trainable low-rank transformations, such that composing the transformed weights no longer yields the invariant exploited by CROSSWEIGHT. We further introduce an attack-aware objective to hinder input reconstruction, together with teacher–student distillation to preserve model utility.

Our main contributions are summarized as follows:

- We reveal a critical privacy vulnerability shared by existing covariant transformation-based private LLM inference schemes. By unifying STIP, CONJFORMER, and AloePri under a common FFN transformation framework, we show that the constraints required to preserve gated FFN and residual computation induce exploitable relations among transformed weights. Building on this, we develop CROSSWEIGHT, which recovers the input-side transformation and reconstructs the private prompt.
- We further propose an irreversible transformation mechanism that breaks the relations exploited by CROSSWEIGHT. The mechanism replaces the original transformations with trainable low-rank transformations and combines an attack-aware objective with teacher–student distillation to hinder prompt reconstruction while mitigating utility degradation.

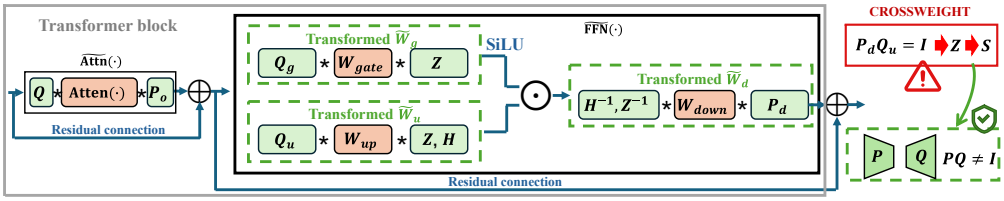


Figure 2: Illustration of the FFN transformation and its structural vulnerability.

- Extensive experiments across multiple LLMs and covariant transformation schemes show that CROSSWEIGHT achieves near-perfect prompt reconstruction, with text-token recovery rates reaching 100%. The proposed irreversible transformation mechanism substantially reduces reconstruction success while providing a more favorable privacy–utility trade-off than existing privacy-preserving LLM inference baselines.

2 PRELIMINARIES AND THREAT MODEL

2.1 MODEL AND NOTATION

We consider a Transformer language model with vocabulary size n, hidden dimension d, and FFN intermediate dimension m. Throughout the paper, symbols with a tilde denote transformed model weights or intermediate representations. For a token sequence S of length s, we denote its intermediate representations by $X \in \mathbb{R}^{s\times d}$. For clarity, we focus on a single Transformer layer in the problem formulation and omit the layer index unless otherwise stated.

Modern LLMs commonly employ gated FFNs (Shazeer, 2020; Grattafiori et al., 2024; Yang et al., 2025), with gate, up, and down projection matrices $W_g, W_u \in \mathbb{R}^{d\times m}$ and $W_d \in \mathbb{R}^{m\times d}$. The FFN computation is given by

$$\mathrm{FFN}(X) = (\sigma(XW_g) \odot XW_u)\, W_d, \tag{1}$$

where $\odot$ denotes the Hadamard product and $\sigma(\cdot)$ is an element-wise activation function, such as SiLU, GELU, or ReLU.

2.2 A UNIFIED FORMULATION OF FFN TRANSFORMATION

The FFN transformation mechanisms used by existing schemes can be expressed in a unified form, as illustrated in Figure 2. First, we define (P, Q) a *compatible transformation pair* if

$$P \in \mathbb{R}^{d\times d'}, \qquad Q \in \mathbb{R}^{d'\times d}, \qquad PQ = I_d, \tag{2}$$

where P and Q are referred to as the key matrix and inverse key matrix, respectively. When $d' = d$, this covers permutation and orthogonal matrices; when $d' > d$, it further accommodates rectangular expansion matrices and their right inverses. Let $Z \in \{0,1\}^{m\times m}$ be a permutation matrix and $H = \mathrm{Diag}(h_1, \ldots, h_m)$, with $h_j \neq 0$, an invertible diagonal scaling matrix. Using subscripts to distinguish key matrices associated with different FFN projections, the transformed weights of FFN can be expressed in the unified form

$$\widetilde{W}_g = Q_g W_g Z, \quad \widetilde{W}_u = Q_u W_u H Z, \quad \widetilde{W}_d = Z^{-1}H^{-1}W_d P_d. \tag{3}$$

This formulation subsumes existing schemes: STIP (Yuan et al., 2026) instantiates the P/Q matrices as permutations with $H = I_m$; CONJFORMER (Yukhimchuk et al., 2026) uses square orthogonal matrices with $H = I_m$; and AloePri (Lin et al., 2026) uses rectangular expansion matrices and their right inverses with $d' > d$, together with a random scaling matrix H.

In the formulation, Q_g and Q_u cancel the incoming transformation from the preceding module to preserve the underlying plaintext gate and up computations, while P_d transforms the FFN output to keep it protected. Consider a transformed FFN input XP_o, where X is the plaintext output of the preceding attention module and P_o is the transformation applied to protect it, as illustrated in Figure 2. Preserving the gate and up computations requires (P_o, Q_g) and (P_o, Q_u) to form compatible transformation pairs, i.e., $P_oQ_g = P_oQ_u = I_d$, so that $XP_oQ_gW_g = XW_g$ and $XP_oQ_uW_u = XW_u$. Similarly, P_d transforms the FFN output and forms a compatible transformation pair with the corresponding input-side transformation of the succeeding module.

The FFN intermediate representations must also remain transformed to protect privacy. At a high level, the element-wise nonlinear activation prevents arbitrary cross-coordinate mixing in the gate branch, restricting its transformation to a permutation Z. The subsequent Hadamard product further requires the gate and up branches to preserve the same coordinate correspondence, forcing the up branch to use the same permutation Z. Since the Hadamard product acts coordinate-wise, the up branch may additionally apply an invertible diagonal scaling H, which is canceled by H^{-1} before the down projection. Hence, the admissible FFN-intermediate transformations take the form in Eq. (3). A formal necessity proof is provided in Appendix C.1. We then obtain

$$\begin{aligned}\widetilde{\mathrm{FFN}}(XP_o) = \left(\sigma(XP_o\widetilde{W}_g) \odot XP_o\widetilde{W}_u\right)\widetilde{W}_d &= \left(\sigma(XW_gZ) \odot XW_uHZ\right)Z^{-1}H^{-1}W_dP_d \\ &= \left(\sigma(XW_g) \odot XW_u\right)W_dP_d = \mathrm{FFN}(X)P_d.\end{aligned} \tag{4}$$

Thus, the transformed FFN exactly preserves the plaintext FFN computation, while its output remains transformed by P_d.

Residual-induced compatibility. The residual connection further constrains the key matrices applied at the FFN input and output. Specifically, the transformed input XP_o is added directly to the transformed FFN output $\mathrm{FFN}(X)P_d$, as in Figure 1. For STIP and CONJFORMER, where P_o and P_d are square and invertible, preserving the residual addition requires $P_d = P_o$, so that $XP_o + \mathrm{FFN}(X)P_d = \big(X + \mathrm{FFN}(X)\big)P_o$. Since $P_oQ_u = I_d$, we therefore have $P_dQ_u = I_d$.

For AloePri, P_o and P_d need not be identical because the key matrices are rectangular. Instead, AloePri adopts a stronger cross-pair compatibility condition: any key matrix P and inverse key matrix Q satisfy $PQ = I_d$, as detailed in Appendix B. Therefore, since P_d is a key matrix and Q_u is an inverse key matrix, AloePri also requires $P_dQ_u = I_d$.

2.3 THREAT MODEL

We consider an honest-but-curious inference server that faithfully follows the prescribed protocol but attempts to recover the user's plaintext prompt from the transformed model weights and intermediate representations available during inference. The secret transformation of model weights is performed outside the inference server, either locally by the user or by a trusted third party, and only the resulting transformed checkpoint is deployed on the server. We focus on the *known-reference-model* setting, where the adversary knows the model architecture, the deployed transformed checkpoint, and the corresponding plaintext checkpoint from which it is derived. This setting is particularly relevant to public or open-weight LLMs, as considered in AloePri (Lin et al., 2026). The adversary does *not* know the secret transformations and does not require paired plaintext–transformed prompts.

3 CROSSWEIGHT: INPUT LEAKAGE FROM CROSS-WEIGHT COMPOSITION

CROSSWEIGHT reconstructs the private prompt in two stages. **Stage I** composes the transformed down and up projection weights to cancel the compatible key matrices P_d and Q_u, removes the diagonal scaling H using the scale-canceling symmetric product, and recovers the FFN permutation Z in Eq. (3) by row matching. **Stage II** uses Z to recover the scheme-specific input transformation and ultimately reconstruct the private prompt.

3.1 STAGE I: BREAKING THE FFN TRANSFORMATION

Step 1: Cross-weight composition. Since each transformed FFN weight in Eq. (3) contains multiple unknown matrices on both sides, directly isolating these matrices from an individual weight is difficult. We therefore exploit the relation of a compatible key pair, $PQ = I_d$, which allows the paired key matrices to cancel under composition. Specifically, we construct the plaintext cross-weight composition $M := W_dW_u$ and its transformed counterpart $B := \widetilde{W}_d\widetilde{W}_u$. Using Eq. (3) and $P_dQ_u = I_d$, we obtain

$$B = Z^{-1}H^{-1}W_dP_dQ_uW_uHZ = Z^{-1}H^{-1}MHZ. \tag{5}$$

Thus, the cross-weight composition cancels P_d and Q_u. Under the *known-reference-model* setting, this leaves only the permutation Z and diagonal scaling H between the known products B and M.

Step 2: Scale-canceling symmetric product. We next eliminate the diagonal scaling matrices H and H^{-1} in Eq. (5). For any $A \in \mathbb{R}^{m\times m}$, we define the scale-canceling symmetric product

$$\mathcal{I}(A) := A \odot A^\top, \qquad [\mathcal{I}(A)]_{ij} = A_{ij}A_{ji}. \tag{6}$$

Lemma 1. *If $B = Z^{-1}H^{-1}MHZ$, where Z is a permutation matrix and H is an invertible diagonal matrix, then $\mathcal{I}(B) = Z^{-1}\mathcal{I}(M)Z$.*

The proof is given in Appendix C.2. By Lemma 1, the symmetric product exactly eliminates the diagonal scaling H, leaving known $\mathcal{I}(B)$ and $\mathcal{I}(M)$ related only by the permutation matrix Z.

Step 3: FFN permutation Z recovery. Let $\mathrm{RowSort}(A)$ sort the entries of each row of A in ascending numerical order. Applying it to $\mathcal{I}(B) = Z^{-1}\mathcal{I}(M)Z$, the right multiplication by the permutation matrix Z only permutes entries within each row and is therefore removed by row-wise sorting, whereas the left multiplication by the permutation matrix Z^{-1} permutes entire rows and is preserved. Specifically, define $\mathrm{Sig}(A) := \mathrm{RowSort}(\mathcal{I}(A)), S_M := \mathrm{Sig}(M), S_B := \mathrm{Sig}(B)$. Then,

$$S_B = \mathrm{RowSort}\big(Z^{-1}\mathcal{I}(M)Z\big) = Z^{-1}\,\mathrm{RowSort}(\mathcal{I}(M)) = Z^{-1}S_M. \tag{7}$$

Thus, recovering Z^{-1} reduces to row matching between S_B and S_M.

For each row a of S_B and row i of S_M, we define the matching cost as $D^{\mathrm{ffn}}_{a,i} := d((S_B)_{a,:}, (S_M)_{i,:})$, where $d(\cdot,\cdot)$ denotes a row-wise distance measure, i.e., the mean squared error (MSE) or the cosine distance. We then solve the one-to-one row-matching problem

$$\widehat{\tau} = \arg\min_{\tau} \sum_{a=1}^{m} D^{\mathrm{ffn}}_{a,\tau(a)}, \tag{8}$$

where $\widehat{\tau}(a) = i$ indicates that row a of S_B is matched to row i of S_M. In practice, we use nearest-neighbor search as an efficient approximation to this assignment. The recovered permutation matrix is therefore given by $(\widehat{Z}^{-1})_{a,i} = \begin{cases} 1, & i = \widehat{\tau}(a), \\ 0, & \text{otherwise.} \end{cases}$ This recovers the permutation matrix Z used in the FFN transformation, completing Stage I of CROSSWEIGHT.

3.2 STAGE II: RECOVERING THE INPUT TRANSFORMATION

We then recover the scheme-specific input transformation below. Since there exists $\widetilde{W}_g = Q_g W_g Z$ in Eq. (3), once Stage I recovers $\widehat{Z}^{-1} = Z^{-1}$, we have $\widetilde{W}_g\widehat{Z}^{-1} = Q_g W_g$. If W_g has full row rank, then $W_g W_g^\dagger = I_d$, and the inverse key matrix Q_g in gate projection can be recovered as

$$\widehat{Q}_g = \widetilde{W}_g\widehat{Z}^{-1}W_g^\dagger = Q_g. \tag{9}$$

STIP and CONJFORMER. For both schemes, the embedding layer is placed on the user side, and input transformation is applied to the plaintext token embeddings before they are exposed to the server, as shown in Figure 1. We denote the plaintext token embeddings by X_e and the corresponding transformed embeddings observed by the server by $\widetilde{X}_e$.

For STIP, the token embeddings are transformed by a permutation matrix P_{in}, such that $\widetilde{X}_e = X_e P_{\mathrm{in}}$; for CONJFORMER, they are transformed by an orthogonal matrix U, such that $\widetilde{X}_e = X_e U$. In both schemes, preserving the underlying plaintext computation requires the input transformation to form a compatible key pair with Q_g, i.e., $P_{\mathrm{in}}Q_g = I_d$ for STIP and $UQ_g = I_d$ for CONJFORMER; see Appendix C.3 for a detailed discussion. Hence, $Q_g = P_{\mathrm{in}}^{-1}$ in STIP and $Q_g = U^{-1}$ in CONJFORMER. Recovering Q_g via Eq. (9) therefore reveals the corresponding input transformation, and the server can recover the private token embeddings as $\widehat{X}_e = \widetilde{X}_e\widehat{Q}_g$. The recovered embeddings can then be matched against the plaintext embedding matrix to identify the private tokens.

AloePri. AloePri instead transformed the input at the vocabulary level. Let $\pi : [n] \to [n]$ denote a secret vocabulary permutation. The client transmits the permuted token indices $\pi^{-1}(S)$, while the server stores the transformed embedding matrix $\widetilde{W}_e = \Pi W_e^\star P_e$, where $W_e^\star = W_e + \alpha_e\mathcal{E}$

is the perturbed embedding matrix, $\mathcal{E} \sim \mathcal{N}(0, \sigma_e^2 I_n \otimes I_d)$, σ_e is the standard variation of W_e, $P_e \in \mathbb{R}^{d \times d'}$ is the key matrix applied to the embedding output and forms a compatible key pair with Q_g, satisfying $P_e Q_g = I_d$ as described in Section 2.2, and $\Pi \in \{0,1\}^{n \times n}$ is the permutation matrix defined by $\Pi_{\pi^{-1}(i),i} = 1$. To reconstruct the private token sequence S from the observed indices $\pi^{-1}(S)$, it suffices to recover π, or equivalently Π.

We then exploit the compatibility relation $P_e Q_g = I_d$ to eliminate P_e through composition by constructing $\widehat{C} := \widetilde{W}_e \widetilde{W}_g \widehat{Z}^{-1}$. Under exact Stage-I recovery, $\widehat{Z}^{-1} = Z^{-1}$, yielding $\widehat{C} = \Pi W_e^\star P_e Q_g W_g = \Pi W_e^\star W_g$. Let $C_0 := W_e W_g$, when $\alpha_e = 0$, we have $W_e^\star = W_e$ and hence $\widehat{C} = \Pi C_0$. Therefore, the rows of $\widehat{C}$ are a permutation of the rows of C_0, and recovering Π reduces to one-to-one row matching between the two matrices using the same assignment as in Eq. (8), with $D_{a,i}^{\text{vocab}} := m^{-1} \big\| (\widehat{C})_{a,:} - (C_0)_{i,:} \big\|_2^2$. With embedding perturbation and imperfect Stage-I recovery, let $R := Z\widehat{Z}^{-1}$, then $\widehat{C} = \Pi C_0 + \Pi C_0 (R - I_m) + \alpha_e \Pi \mathcal{E} W_g R$, where the last two terms capture Stage-I recovery error and embedding perturbation $\mathcal{E}$, respectively. The same row-matching procedure can therefore recover Π provided these perturbations preserve the correct row correspondences. Once Π is recovered, π is known and S follows by applying π to the observed indices $\pi^{-1}(S)$.

4 Private LLM Inference with Irreversible Transformation

Section 3 shows that the compatible key pairs required to preserve inference satisfy $PQ = I_d$ and therefore cancel under cross-weight composition, enabling recovery of the permutation matrix in FFN. To eliminate this vulnerability, we build on AloePri, whose token-index-based design is better suited to practical LLMaaS deployment (Lin et al., 2026), and introduce three components: (i) low-rank transformations that prevent the cancellation exploited in Stage I of CrossWeight; (ii) an attack-aware objective that makes the Stage-II vocabulary row matching less reliable; and (iii) teacher–student distillation that mitigates the resulting utility degradation.

Specifically, we replace each compatible transformation pair (P, Q) with a trainable low-rank transformation pair $\check{P} \in \mathbb{R}^{d \times r}$ and $\check{Q} \in \mathbb{R}^{r \times d}$, where $r < d$. The attack-aware and distillation objectives further jointly optimize these low-rank transformation matrices to reduce prompt-recovery success while preserving model utility. This optimization can be performed offline by the user, after which only the resulting transformed checkpoint is deployed to the inference server. Details of the overall framework are provided in Figure 3 in Appendix D.

4.1 Irreversible Low-Rank Transformation

For the compatible key pair (P_d, Q_u) exploited in Stage I, we replace it with the corresponding low-rank transformation pair $(\check{P}_d, \check{Q}_u)$. Since $\mathrm{rank}(\check{P}_d \check{Q}_u) \le r < d$, we have $\check{P}_d \check{Q}_u \neq I_d$. Therefore, unlike the original compatible key pair, the low-rank transformation pair cannot cancel exactly under cross-weight composition.

Specifically, for each FFN layer, we construct the transformed down and up projection weights as

$$\widetilde{W}_d = Z^{-1} H^{-1} W_d \check{P}_d, \qquad \widetilde{W}_u = \check{Q}_u W_u H Z. \tag{10}$$

Each FFN layer has its own trainable low-rank transformation pair $(\check{P}_d, \check{Q}_u)$; we omit the layer index for simplicity here. The low-rank transformation matrices are incorporated into the transformed weights and not exposed as standalone matrices during inference. Let $K_{d,u} := \check{P}_d \check{Q}_u$. Applying the same cross-weight composition as in Stage I gives $\widetilde{W}_d \widetilde{W}_u = Z^{-1} H^{-1} W_d K_{d,u} W_u H Z$. Thus, the relation in Eq. (5) exploited in Stage I of CrossWeight no longer follows.

4.2 Attack-Aware Row-Matching Objective

Stage II of CrossWeight recovers the vocabulary permutation Π and private inputs via one-to-one row matching between $\widehat{C}$ and C_0 using distance $D_{a,i}^{\text{vocab}}$. While our low-rank transformations already break the exact Stage-I relation for Z, we further improve them against this Stage-II matching to ensure each row in $\widehat{C}$ is no closer to its true corresponding row in C_0 than to other candidates.

Let θ denote all trainable low-rank transformations. For each FFN layer l, we define

$$\widehat{C}_\theta^{(l)} := \widetilde{W}_{e,\theta}\widetilde{W}_{g,\theta}^{(l)}\big(Z^{(l)}\big)^{-1}, \qquad C_0^{(l)} := W_e W_g^{(l)}, \tag{11}$$

where $\widetilde{W}_{e,\theta} = \Pi W_e^\star \check{P}_e$ and $\widetilde{W}_{g,\theta}^{(l)} = \check{Q}_g^{(l)} W_g^{(l)} Z^{(l)}$. Using the Stage-II metric of CROSSWEIGHT, the distance between row a of $\widehat{C}_\theta^{(l)}$ and row i of $C_0^{(l)}$ is $D_{a,i}^{\mathrm{vocab},(l)}(\theta) := m^{-1}\big\|(\widehat{C}_\theta^{(l)})_{a,:} - (C_0^{(l)})_{i,:}\big\|_2^2$. Its distance to the correct plaintext token $\pi(a)$ is therefore $D_a^{+,(l)}(\theta) := D_{a,\pi(a)}^{\mathrm{vocab},(l)}(\theta)$.

To obscure this correct match, we contrast it against the closest incorrect rows. Given training rows $\mathcal{Q} \subseteq [n]$ from $\widehat{C}_\theta^{(l)}$ and candidates $\mathcal{V} \subseteq [n]$ from $C_0^{(l)}$ (with $\pi(a) \in \mathcal{V}, \forall a \in \mathcal{Q}$), let $\mathcal{H}_a^{(l)}$ be the K indices in $\mathcal{V} \setminus \{\pi(a)\}$ minimizing $D_{a,j}^{\mathrm{vocab},(l)}(\theta)$. Defining their average distance as $D_a^{-,(l)}(\theta) := K^{-1}\sum_{j\in\mathcal{H}_a^{(l)}} D_{a,j}^{\mathrm{vocab},(l)}(\theta)$, we minimize $\mathcal{L}_C^{(l)}(\theta) := \frac{1}{|\mathcal{Q}|}\sum_{a\in\mathcal{Q}} \mathrm{softplus}\Big(\frac{D_a^{-,(l)}(\theta) - D_a^{+,(l)}(\theta)}{\tau}\Big)$, where temperature $\tau > 0$ controls smoothness. Minimizing $\mathcal{L}_C^{(l)}$ discourages the correct row from being closer than the nearest incorrect rows, making the true correspondence harder to identify via Stage-II matching. Finally, since Π could be recovered from *any* poorly protected layer, a simple average of $\mathcal{L}_C^{(l)}$ may underemphasize vulnerabilities. Thus, letting $\mathcal{T}(\theta)$ denote the K_L layers with the largest $\mathcal{L}_C^{(l)}(\theta)$, we optimize the worst-case loss: $\mathcal{L}_C(\theta) := \frac{1}{K_L}\sum_{l\in\mathcal{T}(\theta)} \mathcal{L}_C^{(l)}(\theta)$.

4.3 UTILITY RECOVERY VIA DISTILLATION

Because the low-rank pairs introduced above satisfy $\check{P}\check{Q} \neq I_d$, they alter the original model computation and may degrade model utility. We therefore use the plaintext model as the teacher and the transformed model as the student, and optimize θ to align their output distributions.

For a token sequence S, let $p_t(\cdot \mid S_{<t})$ and $p_s(\cdot \mid S_{<t}; \theta)$ denote the teacher's and student's next-token distributions at position t, respectively. We define the distillation objective as the average of the KL divergences in both directions

$$\mathcal{L}_A(\theta) := \frac{1}{2}\mathbb{E}_{S,t}[D_{\mathrm{KL}}(p_t(\cdot \mid S_{<t}) \,\|\, p_s(\cdot \mid S_{<t}; \theta)) + D_{\mathrm{KL}}(p_s(\cdot \mid S_{<t}; \theta) \,\|\, p_t(\cdot \mid S_{<t}))]. \tag{12}$$

Combining the row-matching objective in Section 4.2 with the distillation objective, we optimize

$$\min_\theta \quad \mathcal{L}(\theta) = \mathcal{L}_A(\theta) + \lambda_C \mathcal{L}_C(\theta), \tag{13}$$

where λ_C balances the two objectives.

5 EVALUATION

5.1 EXPERIMENTAL SETUP

Models. Following AloePri (Lin et al., 2026), we evaluate both dense and mixture-of-experts (MoE) LLMs. Specifically, we consider four dense LLMs spanning different scales and architectures: Qwen3-0.6B, Qwen3-1.7B (Yang et al., 2025) and Llama3-8B (Grattafiori et al., 2024). We additionally evaluate the MoE model Qwen3-30B-A3B (Yang et al., 2025).

Datasets and Metrics. For privacy evaluation, we adopt the PUPA benchmark (Siyan et al., 2025), following AloePri (Lin et al., 2026). PUPA contains prompts involving personally identifiable information (PII). We evaluate prompt recovery using four metrics. The Text Token Recovery Success Ratio (TTRSR) measures the percentage of original tokens correctly recovered by the attacker. The PII Recovery Success Ratio (PIIRSR) measures the percentage of personally identifiable information, such as names, addresses, and phone numbers, successfully reconstructed from the protected representations. Recovered Text Cosine Similarity (CosSim) measures the semantic similarity between the recovered text and the original plaintext by computing the cosine similarity between their BGE-M3 embeddings (Chen et al., 2024). BLEU-4 measures the n-gram overlap between the recovered text and the original plaintext.

For utility evaluation, we consider WikiText-2 (Merity et al., 2017) to measure perplexity (PPL), MMLU (Hendrycks et al., 2021) and C-Eval (Huang et al., 2023) to evaluate general knowledge and

Table 1: Recovery performance against STIP, CONJFORMER, and AloePri.

Model	STIP (Yuan et al., 2026)		CONJFORMER (Yukhimchuk et al., 2026)		AloePri (Lin et al., 2026)	
	Z Recovery	P_{in} Recovery	Z Recovery	U MSE	Z Recovery	Π Recovery
Qwen3-1.7B	100.00%	100.00%	100.00%	3.32×10^{-14}	100.00%	98.97%
Llama3-8B	100.00%	100.00%	100.00%	5.69×10^{-14}	100.00%	99.83%
Qwen3-30B-A3B	100.00%	100.00%	100.00%	3.24×10^{-12}	100.00%	99.90%

Table 2: Attack performance against STIP, CONJFORMER, and AloePri. All results are reported in percentage (%).

Model	Attack	STIP			CONJFORMER			AloePri		
		TTRSR	PIIRSR	BLEU-4	TTRSR	PIIRSR	BLEU-4	TTRSR	PIIRSR	BLEU-4
Qwen3-1.7B	**Ours**	**100.00**	**95.58**	**100.00**	**100.00**	**95.58**	**100.00**	**100.00**	**95.58**	**100.00**
	VMA	100.00	95.58	100.00	–	–	–	0.02	0.54	0.00
	IA	33.69	7.42	7.85	32.83	7.46	5.94	1.57	0.97	0.03
	EIA	0.00	0.90	0.00	0.00	1.29	0.00	0.00	1.01	0.00
Llama3-8B	**Ours**	**100.00**	**95.58**	**100.00**	**100.00**	**95.58**	**100.00**	**100.00**	**95.58**	**100.00**
	VMA	100.00	95.58	100.00	–	–	–	0.08	0.17	0.01
	IA	35.46	7.76	8.04	34.69	7.93	7.38	0.00	1.20	0.00
	EIA	0.00	0.99	0.00	0.00	1.07	100.00	0.00	0.73	0.00
Qwen3-30B -A3B	**Ours**	**100.00**	**95.58**	**100.00**	**100.00**	**95.58**	**100.00**	**100.00**	**95.58**	**100.00**
	VMA	100.00	95.58	100.00	–	–	–	0.00	0.77	0.00
	IA	39.92	9.12	12.55	44.33	9.86	15.79	1.05	1.11	0.20
	EIA	0.00	0.95	0.00	0.00	1.07	0.00	0.00	0.86	0.00

reasoning capabilities, and PIQA (Bisk et al., 2020) to evaluate physical commonsense reasoning. We adopt FineWeb (Penedo et al., 2024) to train the proposed low-rank transformations.

Baselines. For privacy attacks, we consider the Vocabulary-Matching Attack (VMA) (Thomas et al., 2025), the Invariant Attack (IA) (Lin et al., 2024), the Embedding Inversion Attack (EIA) (Song & Raghunathan, 2020), the Inversion Model Attack (IMA) (Kugler et al., 2021), and the proposed CROSSWEIGHT. For privacy-preserving inference baselines, we compare our method with representative DP-based approaches, including SANTEXT (Yue et al., 2021), RANTEXT (Tong et al., 2025), and DP-Forward (Du et al., 2023), as well as the state-of-the-art covariant transformation-based method AloePri (Lin et al., 2026). For AloePri, we set the embedding perturbation coefficient α_e to 1.0 as suggested in Lin et al. (2026). Additional details are provided in Appendix E.

5.2 ATTACK EVALUATION ACROSS MODEL ARCHITECTURES

Table 1 reports the two-stage recovery performance of CROSSWEIGHT on existing covariant transformation mechanisms. For each model, we report the best result across all evaluated Transformer layers. CROSSWEIGHT achieves near-perfect recovery across all three existing mechanisms and model architectures, with the FFN permutation Z in Stage I recovered at nearly 100% accuracy in all cases. After recovering Z, Stage II further reconstructs the corresponding input-side secret transformations with high accuracy and negligible error. These results demonstrate that the structural leakage exploited by CROSSWEIGHT generalizes across both different mechanisms and model architectures. Figure 4 in the Appendix further reports layer-wise results on Llama3-8B, where CROSSWEIGHT achieves perfect or near-perfect recovery for most Transformer layers, showing that its effectiveness is not restricted to a particular layer.

We further evaluate end-to-end private prompt reconstruction on the PUPA dataset. Table 2 reports the strongest result of each attack across evaluated layers. While baseline attacks remain effective against STIP or CONJFORMER but largely fail against AloePri, CROSSWEIGHT remains consistently effective across all three mechanisms and model architectures, achieving 100% TTRSR and BLEU-4 together with 95.58% PIIRSR in all evaluated settings. Here, 95.58% corresponds to the empirical upper bound of PIIRSR measured on the plaintext PUPA samples using the adopted PII detector. This confirms that recovery of the secret transformations directly translates into severe end-to-end privacy leakage.

Table 3: Utility and privacy comparison on the Qwen3-1.7B model. The best utility results excluding Plaintext are highlighted in bold.

Method	Generation Performance				Privacy		
	PPL ↓	MMLU ↑	PIQA ↑	C-Eval ↑	Attack	TTRSR (%) ↓	CosSim ↓
Plaintext	8.7018	62.36	76.44	66.27	–	–	–
SANTEXT	523.21	27.99	52.94	27.79	Direct obs.	61.99	0.729
RANTEXT	3611.84	26.02	49.95	25.56	Direct obs.	57.29	0.698
DP-Forward	36827.70	25.12	49.84	24.52	EIA	63.92	0.742
AloePri	24.17	41.18	70.62	39.52	CROSSWEIGHT	100	1.000
					VMA	0.02	0.388
					IA	1.57	0.403
AloePri-Noise+	72801.80	25.74	48.26	26.37	CROSSWEIGHT	36.46	0.556
Proposed	**10.8534**	**54.84**	**73.12**	**56.39**	CROSSWEIGHT	22.24	0.452
					VMA	0.00	0.379
					IA	1.15	0.382
					IMA	0.00	0.378

5.3 COMPARISON WITH PREVIOUS DEFENSE METHODS

Table 3 compares the utility and privacy of the proposed low-rank transformation mechanism with existing privacy-preserving mechanisms, including DP-based and covariant transformation-based methods, on Qwen3-1.7B. Additional results on Llama3-8B and Qwen3-0.6B are reported in Table 4-5 in Appendix F. "Plaintext" denotes the unprotected model and serves as the utility upper bound. For SANTEXT and RANTEXT, "Direct Obs." reports the fraction of tokens that remain unchanged after randomized replacement and can therefore be directly recovered by the server. As shown in the table, SANTEXT, RANTEXT, and DP-Forward require stronger perturbations to provide meaningful privacy protection, at the cost of sharply increased perplexity and near-random accuracy on downstream benchmarks. Nevertheless, their privacy protection remains considerably weaker than that of the proposed method, indicating an unfavorable privacy–utility trade-off.

AloePri preserves substantially higher model utility than the DP-based baselines, but provides essentially no protection against the proposed attack: CROSSWEIGHT achieves 100% TTRSR with a cosine similarity of 1.000 on examined model scales. Since AloePri injects Gaussian noise into the embedding matrix as part of its privacy mechanism, we additionally evaluate an *AloePri-Noise+* variant, in which the perturbation coefficient α_e is increased to reduce the attack success rate. However, the resulting privacy improvement comes at a substantial utility cost. For example, on Qwen3-1.7B, AloePri-Noise+ reduces the CROSSWEIGHT TTRSR to 36.46%, which remains higher than the 22.24% achieved by our method, while its perplexity increases to $72{,}801.80$ and downstream-task performance largely collapses.

In contrast, the proposed low-rank transformation mechanism always achieves the strongest overall privacy–utility trade-off. Although it introduces slight utility degradation relative to the plaintext model, the degradation is substantially smaller than that of the competing privacy-preserving baselines, while the effectiveness of CROSSWEIGHT, VMA, and IA is consistently reduced. Similar trends are observed on the Llama3-8B and Qwen3-0.6B in Table 4-5 in Appendix F. These results demonstrate that the proposed mechanism mitigates the privacy leakage exploited by CROSSWEIGHT while maintaining a substantially better utility–privacy balance than existing privacy-preserving baselines.

6 CONCLUSION

We identify a critical privacy vulnerability in existing covariant transformation schemes for private LLM inference. By exploiting the constraints induced by gated FFN and residual connections, CROSSWEIGHT recovers secret transformations in the transformed FFN by composing transformed weights and then reconstructs private prompts with near-perfect accuracy across multiple schemes and model architectures. To mitigate this leakage, we propose an irreversible transformation mechanism with low-rank transformations, attack-aware training, and utility-preserving distillation. Experiments show that it substantially reduces reconstruction effectiveness while achieving a more favorable privacy–utility trade-off than existing baselines.

AI USE STATEMENT

In this work, we did not use generative AI tools for any tasks requiring mandatory disclosure. In particular, generative AI tools were not used to develop theoretical models or conceptual frameworks, formulate or prove mathematical claims, design or provide feedback on research methodology or experiments, implement methods, or interpret results. The remaining tasks requiring mandatory disclosure are not applicable to this work.

Additionally, we used generative AI tools for tasks with recommended disclosure, specifically to edit the manuscript to improve readability and to assist with editing and debugging author-written experimental code. We have reviewed all AI-assisted work. All AI-assisted text was manually reviewed and revised by the authors to preserve the intended meaning, and all AI-assisted code modifications were manually inspected and tested by the authors to verify correctness. We take responsibility for the final content of this work, including text, claims, and code produced with the aid of generative AI.

REFERENCES

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7432–7439, 2020. doi: 10.1609/aaai.v34i05.6239.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 2318–2335, Bangkok, Thailand, 2024. Association for Computational Linguistics.

Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. In *Advances in Cryptology – ASIACRYPT 2017*, pp. 409–437. Springer, 2017. doi: 10.1007/978-3-319-70694-8_15.

Tian Dong, Yan Meng, Shaofeng Li, Guoxing Chen, Zhen Liu, and Haojin Zhu. Depth gives a false sense of privacy: LLM internal states inversion. In *34th USENIX Security Symposium (USENIX Security 25)*, pp. 1629–1648, Seattle, WA, August 2025. USENIX Association. ISBN 978-1-939133-52-6.

Ye Dong, Wen jie Lu, Yancheng Zheng, Haoqi Wu, Derun Zhao, Jin Tan, Zhicong Huang, Cheng Hong, Tao Wei, and Wenguang Chen. PUMA: Secure inference of LLaMA-7b in five minutes. *arXiv preprint arXiv:2307.12533*, 2023.

Minxin Du, Xiang Yue, Sherman S. M. Chow, Tianhao Wang, Chenyu Huang, and Huan Sun. DP-Forward: Fine-tuning and inference on language models with differential privacy in forward pass. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2665–2679, 2023. doi: 10.1145/3576915.3616592.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. doi: 10.48550/arXiv.2407.21783.

Yujie Gu, Richeng Jin, Xiaoyu Ji, Yier Jin, and Wenyuan Xu. Differentially private and communication efficient large language model split inference via stochastic quantization and soft prompt. *arXiv preprint arXiv:2602.11513*, 2026.

Kanav Gupta, Neha Jawalkar, Ananta Mukherjee, Nishanth Chandran, Divya Gupta, Ashish Panwar, and Rahul Sharma. SIGMA: Secure GPT inference with function secret sharing. *Proceedings on Privacy Enhancing Technologies*, 2024(4):61–79, 2024. doi: 10.56553/popets-2024-0107.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, Yao Fu, Maosong Sun, and Junxian He. C-Eval: A multi-level multi-discipline chinese evaluation suite for foundation models. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Kai Kugler, Simon Münker, Johannes Höhmann, and Achim Rettinger. Invbert: Reconstructing text from contextualized word embeddings by inverting the bert pipeline. *arXiv preprint arXiv:2109.10104*, 2021.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626. ACM, 2023. doi: 10.1145/3600006.3613165.

Dacheng Li, Rulin Shao, Hongyi Wang, Han Guo, Eric P. Xing, and Hao Zhang. MPCFormer: Fast, performant and private transformer inference with MPC. In *The Eleventh International Conference on Learning Representations*, 2023.

Yu Lin, Qizhi Zhang, Quanwei Cai, Jue Hong, Wu Ye, Huiqi Liu, and Bing Duan. An inversion attack against obfuscated embedding matrix in language model inference. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 2100–2104. Association for Computational Linguistics, November 2024.

Yu Lin, Qizhi Zhang, Wenqiang Ruan, Daode Zhang, Jue Hong, Ye Wu, Hanning Xia, Yunlong Mao, and Sheng Zhong. Towards privacy-preserving LLM inference via covariant obfuscation. *arXiv preprint arXiv:2603.01499*, 2026. Technical report, version 2.

Wenjie Lu, Zhicong Huang, Zhen Gu, Jingyu Li, Jian Liu, Cheng Hong, Kui Ren, Tao Wei, and WenGuang Chen. BumbleBee: Secure two-party inference framework for large transformers. In *Network and Distributed System Security Symposium*. Internet Society, 2025.

Jinglong Luo, Guanzhong Chen, Yehong Zhang, Shiyu Liu, Hui Wang, Yue Yu, Xun Zhou, Yuan Qi, and Zenglin Xu. CENTAUR: Bridging the impossible trinity of privacy, efficiency, and performance in privacy-preserving transformer inference. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 22751–22770, Vienna, Austria, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.1109.

Peihua Mai, Ran Yan, Zhe Huang, Youjia Yang, and Yan Pang. Split-and-denoise: Protect large language model inference with local differential privacy. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 34281–34302. PMLR, 2024.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, April 2017.

Jungho Moon, Dongwoo Yoo, Xiaoqian Jiang, and Miran Kim. THOR: Secure transformer inference with homomorphic encryption. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, 2025. doi: 10.1145/3719027.3765150.

John Morris, Volodymyr Kuleshov, Vitaly Shmatikov, and Alexander Rush. Text embeddings reveal (almost) as much as text. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12448–12460, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.765.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The FineWeb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Ying Sheng, Lianmin Zheng, Binhang Yuan, Zhuohan Li, Max Ryabinin, Beidi Chen, Percy Liang, Christopher Ré, Ion Stoica, and Ce Zhang. Flexgen: High-throughput generative inference of large language models with a single gpu. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 31094–31116. PMLR, 2023.

Li Siyan, Vethavikashini Chithrra Raghuram, Omar Khattab, Julia Hirschberg, and Zhou Yu. PAPILLON: Privacy preservation from Internet-based and local language model ensembles. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3371–3390, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.naacl-long.173.

Congzheng Song and Ananth Raghunathan. Information leakage in embedding models. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, CCS '20, pp. 377–390, New York, NY, USA, 2020. Association for Computing Machinery. doi: 10.1145/3372297.3417270.

Yifan Tan, Cheng Tan, Zeyu Mi, and Haibo Chen. PipeLLM: Fast and confidential large language model services with speculative pipelined encryption. In *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1*, pp. 843–857. Association for Computing Machinery, 2025. doi: 10.1145/3669940.3707224.

Rahul Krishna Thomas, Louai Zahran, Erica Choi, Akilesh Potti, Micah Goldblum, and Arka Pal. Hidden no more: Attacking and defending private third-party LLM inference. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 59434–59469. PMLR, 2025.

Meng Tong, Kejiang Chen, Jie Zhang, Yuang Qi, Weiming Zhang, Nenghai Yu, Tianwei Zhang, and Zhikun Zhang. InferDPT: Privacy-preserving inference for closed-box large language models. *IEEE Transactions on Dependable and Secure Computing*, 22(5):4625–4640, Sep.–Oct. 2025. doi: 10.1109/TDSC.2025.3550389.

Tianshi Xu, Wen jie Lu, Jiangrui Yu, Yi Chen, Chenqi Lin, Runsheng Wang, and Meng Li. Breaking the layer barrier: Remodeling private transformer inference with hybrid CKKS and MPC. In *34th USENIX Security Symposium*, pp. 2653–2672, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. doi: 10.48550/arXiv.2505.09388.

Mu Yuan, Lan Zhang, Yihang Cheng, Miao-Hui Song, Guoliang Xing, and Xiang-Yang Li. STIP: Three-party privacy-preserving and lossless inference for large transformers in production. In *Network and Distributed System Security Symposium*. Internet Society, 2026.

Xiang Yue, Minxin Du, Tianhao Wang, Yaliang Li, Huan Sun, and Sherman S. M. Chow. Differential privacy for text analytics via natural text sanitization. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3853–3866. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.findings-acl.337.

Alexander Yukhimchuk, Andrey Shulga, Mladen Kolar, and Martin Takac. Privacy from symmetry: Orthogonally equivariant transformers for LLM inference. *arXiv preprint arXiv:2606.16461*, 2026.

Fei Zheng, Chaochao Chen, Zhongxuan Han, and Xiaolin Zheng. PermLLM: Private inference of large language models within 3 seconds under WAN. *arXiv preprint arXiv:2405.18744*, 2024.

A Related Work

A.1 Privacy-Preserving LLM Inference

Cryptographic and trusted-hardware approaches. Cryptographic approaches protect private inputs and intermediate representations through primitives such as homomorphic encryption (HE), secure multiparty computation (MPC), and secret sharing. SIGMA (Gupta et al., 2024) combines secret sharing with function secret sharing for secure GPT inference, while BumbleBee (Lu et al., 2025) and MPCFormer (Li et al., 2023) design specialized secure protocols for Transformer inference. PermLLM (Zheng et al., 2024) further employs secure random permutations to reduce the cost of evaluating nonlinear operations, together with secret sharing and HE. Similarly, CENTAUR (Luo et al., 2025) combines random permutation with secure computation to improve the efficiency of private Transformer inference. Although these methods provide cryptographic protection under their respective security assumptions, they typically incur substantial computational and communication overhead. Trusted-hardware approaches instead rely on hardware-isolated execution environments. For example, PipeLLM (Tan et al., 2025) leverages GPU confidential computing together with speculative pipelined encryption for confidential LLM serving. In contrast, our setting requires no trusted hardware and directly deploys an obfuscated checkpoint to an honest-but-curious server.

Perturbation-based approaches. Another line of work protects inference inputs by perturbing either discrete tokens or their continuous representations. SANTEXT (Yue et al., 2021) applies differentially private text sanitization, while RANTEXT (Tong et al., 2025) perturbs input tokens through a local-DP mechanism for black-box LLM inference. At the representation level, DP-Forward (Du et al., 2023) injects calibrated noise into embeddings or intermediate hidden states during the forward pass. Split-and-Denoise (Mai et al., 2024) perturbs representations transmitted from the client and employs a denoising module to mitigate the resulting utility degradation. DEL (Gu et al., 2026) further combines embedding projection with server-side soft prompts to compensate for the utility degradation caused by privacy-preserving perturbation. However, these perturbation-based methods generally face an inherent privacy–utility trade-off, as stronger perturbations for improved privacy can lead to substantial degradation in model utility.

Covariant Transformation-based Obfuscation. Covariant transformation provides an efficient approach to privacy-preserving LLM inference in terms of both computational overhead and model utility. The core idea is to jointly transform private inputs, intermediate representations, and the corresponding model weights using secret transformations, thereby concealing the plaintext inputs and representations from the inference server while preserving the underlying plaintext computation. STIP (Yuan et al., 2026) applies secret feature-space permutations and correspondingly transforms the model parameters to enable efficient private Transformer inference. ConjFormer (Yukhimchuk et al., 2026) constructs an orthogonally equivariant Transformer whose computation can be performed directly in a secretly transformed feature space. AloePri (Lin et al., 2026) further combines vocabulary permutation, noise injection, and compatible linear transformations while remaining compatible with existing LLM inference engines by permuting token indices rather than token embeddings. Compared with perturbation-based methods, they can preserve near-lossless model utility while also demonstrating empirical robustness against existing reconstruction attacks.

A.2 Privacy Attacks on LLM Inference

Representation-inversion attacks. A substantial body of work studies the reconstruction of private text from embeddings or intermediate hidden states. Simple nearest-neighbor attacks recover tokens by matching observed representations against a known embedding table (Tong et al., 2025), whereas optimization-based attacks search directly for inputs whose representations match the observed ones. InvBERT (Kugler et al., 2021) trains an inversion model to reconstruct text from contextualized word embeddings. Vec2Text (Morris et al., 2023) further formulates embedding inversion as an iterative generation problem, repeatedly refining a candidate reconstruction using the discrepancy between its re-embedded representation and the target embedding. More recently, the internal-state inversion attack of Dong et al. (Dong et al., 2025) demonstrates that even representations from deep LLM layers can retain substantial information about the original input. These attacks can reconstruct the text corresponding to observed hidden representations. However, they

are not applicable to transformation-based obfuscation, where representations are transformed by secret transformations. For example, under a secret permutation, inverting the observed hidden states would recover the corresponding permuted tokens rather than the true plaintext tokens.

Transformation-recovery attacks. A complementary line of work exploits structural relationships introduced by the transformation mechanism itself. The Invariant Attack (IA) (Lin et al., 2024) exploits statistical or algebraic quantities that remain unchanged under embedding transformation to recover the correspondence between plaintext and transformed representations. The Vocabulary-Matching Attack (VMA) (Thomas et al., 2025) exploits relationships between known plaintext model components and their transformed counterparts to recover hidden vocabulary permutations. AloePri (Lin et al., 2026) further considers variants of invariant-based attacks that exploit gate and attention structures when evaluating the security of covariant transformation.

B Compatibility of AloePri's Key Matrices

AloePri constructs a family of rectangular expansion and contraction key matrices from a shared set of base matrices. Let h denote the expansion parameter, such that

$$d' = d + 2h. \tag{14}$$

The construction first generates an invertible matrix $B \in \mathbb{R}^{d\times d}$, two auxiliary matrices

$$E \in \mathbb{R}^{d\times h}, \qquad F \in \mathbb{R}^{h\times d}, \tag{15}$$

and an orthogonal mixing matrix $R \in \mathbb{R}^{d'\times d'}$ satisfying $RR^\top = I_{d'}$.

For each expansion-key index ρ, AloePri samples $C_\rho \in \mathbb{R}^{d\times h}$ satisfying

$$C_\rho F = 0 \tag{16}$$

and constructs

$$P_\rho = [B \quad C_\rho \quad E]\, R \in \mathbb{R}^{d\times d'}. \tag{17}$$

Equivalently, the rows of C_ρ are sampled from $\mathrm{null}(F^\top)$.

Similarly, for each contraction-key index σ, it samples $D_\sigma \in \mathbb{R}^{h\times d}$ satisfying

$$ED_\sigma = 0 \tag{18}$$

and constructs

$$Q_\sigma = R^\top \begin{bmatrix} B^{-1} \\ F \\ D_\sigma \end{bmatrix} \in \mathbb{R}^{d'\times d}. \tag{19}$$

Equivalently, the columns of D_σ are sampled from $\mathrm{null}(E)$.

For any P_ρ and Q_σ generated from these shared base matrices, we have

$$P_\rho Q_\sigma = [B \quad C_\rho \quad E]\, RR^\top \begin{bmatrix} B^{-1} \\ F \\ D_\sigma \end{bmatrix} = BB^{-1} + C_\rho F + ED_\sigma = I_d. \tag{20}$$

Importantly, this identity holds for every expansion key P_ρ and contraction key Q_σ generated from the same shared base matrices, rather than only for a uniquely paired (P_ρ, Q_ρ).

C Additional Proofs

C.1 Necessity of the FFN Intermediate Transformation Form

To characterize the constraints that a transformation family must satisfy to preserve gated FFN computation, we consider SiLU-based SwiGLU, which is widely adopted in modern LLMs. Any transformation family intended to support a broad class of gated FFNs must, in particular, preserve this commonly used instance. We show that, within the class of invertible linear transformations,

the intermediate-space transformation form in Eq. (3) is not merely sufficient but necessary for exact preservation of the SiLU-based gated FFN computation.

Let

$$\phi(t) := \mathrm{SiLU}(t) = t\,\sigma(t) = \frac{t}{1+e^{-t}},$$

where $\sigma(t) = 1/(1+e^{-t})$ is the sigmoid function, and let $\phi(\cdot)$ act element-wise on vectors. To isolate the intermediate-space constraint, consider arbitrary gate- and up-branch representations $a, b \in \mathbb{R}^{1\times m}$. Let $A_g, A_u, A_d \in \mathbb{R}^{m\times m}$ be invertible linear transformations applied after the gate projection, after the up projection, and before the down projection, respectively. Exact preservation of the gated intermediate computation requires

$$(\phi(aA_g) \odot bA_u)\, A_d = \phi(a) \odot b, \qquad \forall\, a, b \in \mathbb{R}^{1\times m}. \tag{21}$$

Theorem 1 (Necessary Form of Gated-FFN-Preserving Transformations)**.** *Suppose A_g, A_u, A_d are invertible and satisfy Eq. (21) for all $a, b \in \mathbb{R}^{1\times m}$. Then there exist a permutation matrix $Z \in \{0,1\}^{m\times m}$ and an invertible diagonal matrix $H = \mathrm{Diag}(h_1, \ldots, h_m)$ such that*

$$A_g = Z, \qquad A_u = HZ, \qquad A_d = Z^{-1}H^{-1}. \tag{22}$$

Proof. For row vectors p and q, we have $p \odot q = q\,\mathrm{Diag}(p)$. Therefore, Eq. (21) can be rewritten as

$$bA_u\,\mathrm{Diag}(\phi(aA_g))\,A_d = b\,\mathrm{Diag}(\phi(a))\,.$$

Since this equality holds for every b,

$$A_u\,\mathrm{Diag}(\phi(aA_g))\,A_d = \mathrm{Diag}(\phi(a))\,, \qquad \forall\, a. \tag{23}$$

Step 1: Characterizing the gate-side transformation. For SiLU,

$$\phi'(0) = \frac{1}{2}, \qquad \phi''(0) = \frac{1}{2},$$

so both derivatives are nonzero. Let r_j denote the j-th row of A_g. Differentiating Eq. (23) with respect to a_j and evaluating at $a = 0$ gives

$$\phi'(0)A_u\,\mathrm{Diag}(r_j)A_d = \phi'(0)E_{jj},$$

where E_{jj} contains a single nonzero entry at (j, j). Since $\phi'(0) \neq 0$,

$$A_u\,\mathrm{Diag}(r_j)A_d = E_{jj}. \tag{24}$$

Differentiating twice with respect to a_j gives

$$\phi''(0)A_u\,\mathrm{Diag}(r_j \odot r_j)A_d = \phi''(0)E_{jj}.$$

Since $\phi''(0) \neq 0$,

$$A_u\,\mathrm{Diag}(r_j \odot r_j)A_d = E_{jj}. \tag{25}$$

Comparing Eqs. (24) and (25) and using the invertibility of A_u and A_d yields

$$r_j \odot r_j = r_j. \tag{26}$$

Hence every entry of A_g is either 0 or 1.

For two distinct coordinates $j \neq k$, taking the mixed second derivative of Eq. (23) with respect to a_j and a_k gives

$$\phi''(0)A_u\,\mathrm{Diag}(r_j \odot r_k)A_d = 0.$$

Again using $\phi''(0) \neq 0$ and the invertibility of A_u and A_d, we obtain

$$r_j \odot r_k = 0, \qquad j \neq k. \tag{27}$$

Thus, distinct rows of A_g have disjoint supports. Because A_g is invertible, every row is nonzero. The m nonzero rows therefore have pairwise disjoint supports over m coordinates, implying that every row and every column contains exactly one nonzero entry. Together with Eq. (26), this gives

$$A_g = Z \tag{28}$$

for some permutation matrix Z.

Step 2: Characterizing the up- and down-side transformations. Substituting $A_g = Z$ into Eq. (23) and using the element-wise property

$$\phi(aZ) = \phi(a)Z$$

gives

$$A_u \operatorname{Diag}(\phi(a)Z)\, A_d = \operatorname{Diag}(\phi(a))\,.$$

For a permutation matrix Z,

$$\operatorname{Diag}(vZ) = Z^{-1} \operatorname{Diag}(v) Z.$$

Hence

$$\left(A_u Z^{-1}\right) \operatorname{Diag}(\phi(a))\, (ZA_d) = \operatorname{Diag}(\phi(a))\,. \tag{29}$$

Since $\phi'(0) \neq 0$, the inverse function theorem implies that SiLU is locally invertible around the origin. Therefore, $\operatorname{Diag}(\phi(a))$ contains an open neighborhood of the zero matrix within the vector space of diagonal matrices. Because both sides of Eq. (29) depend linearly on this diagonal matrix, the identity extends to every diagonal matrix D:

$$BDC = D, \qquad \text{for every diagonal } D, \tag{30}$$

where

$$B := A_u Z^{-1}, \qquad C := ZA_d.$$

Taking $D = E_{ii}$ in Eq. (30) gives

$$Be_i e_i^{\mathsf{T}} C = e_i e_i^{\mathsf{T}}.$$

The left-hand side is the outer product of the i-th column of B and the i-th row of C. Since the right-hand side has support only at (i, i), both vectors can have support only on coordinate i. Because B and C are invertible, there exists a nonzero scalar h_i such that

$$Be_i = h_i e_i, \qquad e_i^{\mathsf{T}} C = h_i^{-1} e_i^{\mathsf{T}}.$$

Thus,

$$B = H, \qquad C = H^{-1},$$

where $H = \operatorname{Diag}(h_1, \ldots, h_m)$ is invertible. Recalling the definitions of B and C, we obtain

$$A_u = HZ, \qquad A_d = Z^{-1}H^{-1}.$$

Together with $A_g = Z$, this proves Eq. (22). □

C.2 Proof of Lemma 1

Proof. Let

$$H = \operatorname{Diag}(h_1, \ldots, h_m).$$

Then

$$[H^{-1}MH]_{ij} = \frac{h_j}{h_i} M_{ij}. \tag{31}$$

Hence,

$$[\mathcal{I}(H^{-1}MH)]_{ij} = \frac{h_j}{h_i} M_{ij} \frac{h_i}{h_j} M_{ji} = M_{ij}M_{ji} = [\mathcal{I}(M)]_{ij}. \tag{32}$$

Therefore,

$$\mathcal{I}(H^{-1}MH) = \mathcal{I}(M). \tag{33}$$

Moreover, simultaneous row and column permutations are equivariant under $\mathcal{I}$, i.e.,

$$\mathcal{I}(Z^{-1}AZ) = Z^{-1}\mathcal{I}(A)Z. \tag{34}$$

Applying the above two identities to $B = Z^{-1}H^{-1}MHZ$ yields

$$\mathcal{I}(B) = Z^{-1}\mathcal{I}(M)Z, \tag{35}$$

which proves Lemma 1. □

C.3 Input–FFN Compatibility in STIP and ConjFormer

We detail how the input transformation and FFN weight transformation in STIP and ConjFormer induce the compatibility relations used in Section 3.2.

STIP. STIP applies a feature-space permutation π to the token embeddings before they are sent to the server. Specifically, given the plaintext token embeddings X_e, the server observes

$$\widetilde{X}_e = X_e\pi. \tag{36}$$

For the gated FFN, STIP transforms the gate, up, and down projections as

$$\widetilde{W}_g = \pi^\mathsf{T} W_g Z, \qquad \widetilde{W}_u = \pi^\mathsf{T} W_u Z, \qquad \widetilde{W}_d = Z^{-1} W_d \pi, \tag{37}$$

where Z corresponds to the FFN-intermediate permutation used by STIP. Therefore, under our notation,

$$P_{\text{in}} = \pi, \qquad Q_g = \pi^\mathsf{T}.$$

Since π is a permutation matrix, $\pi^\mathsf{T} = \pi^{-1}$, and hence

$$P_{\text{in}} Q_g = \pi\pi^\mathsf{T} = I_d. \tag{38}$$

Thus, the input permutation in STIP is exactly the inverse of the input-side key Q_g applied to the gate projection.

ConjFormer. ConjFormer applies a secret orthogonal transformation to the token embeddings and conjugates the Transformer weights accordingly. To distinguish the notation in its original formulation from ours, let $R \in \mathbb{R}^{d\times d}$ denote the secret orthogonal matrix. The transmitted embeddings are

$$\widetilde{X}_e = X_e R^\mathsf{T}. \tag{39}$$

For a gated FFN, ConjFormer uses the PyTorch convention XW^T and transforms the gate, up, and down projections as

$$\widehat{W}_{\text{gate}} = Z^\mathsf{T} W_{\text{gate}} R^\mathsf{T}, \quad \widehat{W}_{\text{up}} = Z^\mathsf{T} W_{\text{up}} R^\mathsf{T}, \quad \widehat{W}_{\text{down}} = R W_{\text{down}} Z, \tag{40}$$

where Z is the permutation applied in the FFN intermediate space.

Our formulation instead writes linear projections as XW. Accordingly, we define

$$W_g = W_{\text{gate}}^\mathsf{T}, \qquad W_u = W_{\text{up}}^\mathsf{T}, \qquad W_d = W_{\text{down}}^\mathsf{T},$$

and similarly

$$\widetilde{W}_g = \widehat{W}_{\text{gate}}^\mathsf{T}, \qquad \widetilde{W}_u = \widehat{W}_{\text{up}}^\mathsf{T}, \qquad \widetilde{W}_d = \widehat{W}_{\text{down}}^\mathsf{T}.$$

Transposing Eq. (40) gives

$$\widetilde{W}_g = R W_g Z, \quad \widetilde{W}_u = R W_u Z, \quad \widetilde{W}_d = Z^{-1} W_d R^\mathsf{T}, \tag{41}$$

where $Z^\mathsf{T} = Z^{-1}$ since Z is a permutation matrix.

Meanwhile, the input transformation in our notation is

$$U := R^\mathsf{T}.$$

Since R is orthogonal,

$$R = (R^\mathsf{T})^{-1} = U^{-1}.$$

Therefore, Eq. (41) becomes

$$\widetilde{W}_g = U^{-1} W_g Z, \quad \widetilde{W}_u = U^{-1} W_u Z, \quad \widetilde{W}_d = Z^{-1} W_d U. \tag{42}$$

Comparing these expressions with Eq. (3), we identify

$$Q_g = Q_u = U^{-1}, \qquad P_d = U, \qquad H = I_m.$$

Consequently,

$$U Q_g = U Q_u = I_d.$$

Hence, in both STIP and ConjFormer, the transformation applied to the transmitted token embeddings is the inverse of the input-side key Q_g appearing in the FFN projection.

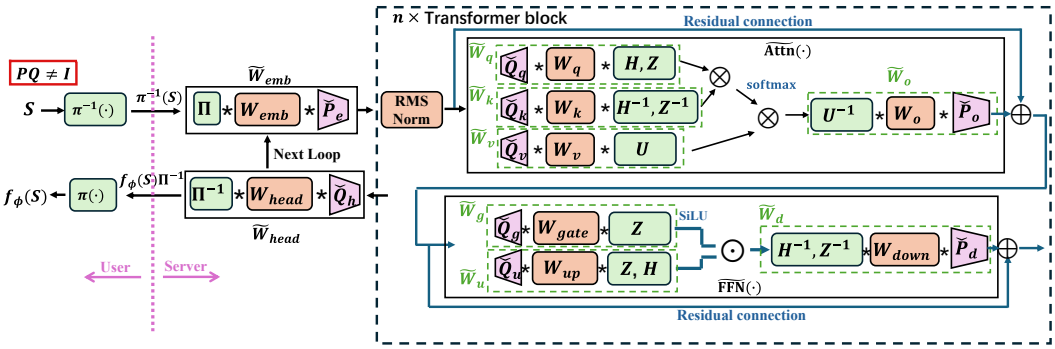


Figure 3: Detailed architecture of the proposed irreversible transformation framework. The framework follows the token-index-based inference interface structure of AloePri (Lin et al., 2026), while replacing the original compatible transformation matrices with trainable low-rank transformations. The low-rank transformations are folded into the deployed transformed model weights, while the client only applies the secret vocabulary permutation during online inference.

D Detailed Framework Architecture

As illustrated in Figure 3, the proposed mechanism largely follows the token-index-based deployment and module-wise transformation structure of AloePri (Lin et al., 2026). The main difference is that the original compatible transformation pairs satisfying $PQ = I_d$ are replaced by trainable low-rank transformations $\check{P} \in \mathbb{R}^{d\times r}$ and $\check{Q} \in \mathbb{R}^{r\times d}$ with $r < d$. These transformations are folded into the corresponding model weights and jointly optimized by the attack-aware and distillation objectives described in Section 4. Below, we provide additional details on how the transformations are integrated with the model components.

Embedding and model head. We retain AloePri's vocabulary-level interface so that the client only performs a lightweight secret permutation during online inference. Let Π denote the permutation matrix associated with the secret vocabulary permutation π. Unlike AloePri, we do not inject additional noise into the embedding or model head. AloePri introduces such perturbations primarily to reduce the effectiveness of vocabulary-matching attacks such as VMA, whereas our trainable low-rank transformations directly break the compatibility condition $PQ = I$ and thereby provide effective resistance to this class of attacks. Details of the VMA attack are provided in Appendix E. The transformed weights of the embedding and head layers are constructed as

$$\widetilde{W}_e = \Pi W_e \check{P}_e, \qquad \widetilde{W}_h = \check{Q}_h W_h \Pi^{-1}, \tag{43}$$

where $\check{P}_e$ and $\check{Q}_h$ are the corresponding trainable low-rank transformations. During online inference, the client maps plaintext token indices through the secret permutation $\pi^{-1}(\cdot)$ before transmission and applies the inverse mapping $\pi(\cdot)$ to the returned token indices. Hence, no embedding computation is required on the client side.

Attention transformation. As illustrated in Figure 3, similar to the attention-specific design of AloePri (Lin et al., 2026), we apply paired diagonal scaling and permutation transformations to the query and key branches, and an invertible transformation and its inverse to the value and output branches, respectively. Specifically, the transformed attention weights are constructed as

$$\widetilde{W}_q = \check{Q}_q W_q H Z, \quad \widetilde{W}_k = \check{Q}_k W_k H^{-1} Z, \quad \widetilde{W}_v = \check{Q}_v W_v U, \quad \widetilde{W}_o = U^{-1} W_o \check{P}_o, \tag{44}$$

where H is an invertible diagonal scaling matrix, Z is a permutation matrix, U is an invertible transformation in the value space, and $\check{Q}_q, \check{Q}_k, \check{Q}_v, \check{P}_o$ are the corresponding trainable low-rank transformations.

To clarify the role of the attention-specific transformations, we temporarily omit the low-rank input- and output-side transformations and define

$$W'_q = W_q H Z, \qquad W'_k = W_k H^{-1} Z, \qquad W'_v = W_v U, \qquad W'_o = U^{-1} W_o. \tag{45}$$

For an input representation X, the transformed query–key product satisfies

$$(XW'_q)(XW'_k)^\mathsf{T} = XW_qHZ\left(XW_kH^{-1}Z\right)^\mathsf{T} = XW_qHZZ^\mathsf{T}H^{-1}W_k^\mathsf{T}X^\mathsf{T} = XW_qW_k^\mathsf{T}X^\mathsf{T}, \tag{46}$$

where we use $ZZ^\mathsf{T} = I$ and the fact that H is diagonal. Similarly, the value–output transformations cancel as

$$(XW'_v)W'_o = XW_vUU^{-1}W_o = XW_vW_o. \tag{47}$$

Hence, ignoring the low-rank boundary transformations, these paired transformations protect the intermediate query, key, and value representations while exactly preserving the plaintext attention computation.

FFN and residual connections. The FFN follows the irreversible low-rank construction introduced in Section 4.1. Specifically,

$$\widetilde{W}_g = \check{Q}_gW_gZ, \qquad \widetilde{W}_u = \check{Q}_uW_uHZ, \qquad \widetilde{W}_d = Z^{-1}H^{-1}W_d\check{P}_d, \tag{48}$$

where Z and H denote the FFN permutation and diagonal scaling transformations, respectively. The outgoing transformations of the attention and FFN modules are aligned with their residual branches so that both operands of each residual addition remain in the same transformed representation space. Consequently, transformed representations can propagate through successive Transformer blocks without exposing their plaintext counterparts.

E Experimental Details

Following AloePri (Lin et al., 2026), we implement the attack baselines as follows.

- **Vocabulary-Matching Attack (VMA).** VMA (Thomas et al., 2025) considers a known plaintext matrix X and its transformed counterpart

  $$Y = Z_1XZ_2, \tag{49}$$

  where Z_1 and Z_2 are unknown permutation matrices. Since a column permutation only reorders the entries within each row, row-wise sorting eliminates Z_2:

  $$\mathrm{RowSort}(Y) = Z_1\,\mathrm{RowSort}(X). \tag{50}$$

  VMA then recovers Z_1 by nearest-neighbor matching between the sorted row signatures of X and Y.

  For STIP, using the FFN transformations in Eq. (37), the up and down projections can be composed as

  $$\widetilde{W}_u\widetilde{W}_d = \pi^\mathsf{T}W_uW_d\pi. \tag{51}$$

  This has exactly the form of Eq. (49), with $X = W_uW_d$, $Y = \widetilde{W}_u\widetilde{W}_d$, and $Z_1 = \pi^\mathsf{T}$. Since both the plaintext and transformed weights are available under our threat model, VMA can recover π, which is also the permutation applied to the input embeddings in Eq. (36), and thereby invert the input transformation.

  For ConjFormer, however, the feature-space transformations are dense orthogonal matrices rather than permutations. Consequently, row-wise sorting does not eliminate the unknown right-side transformation in general, i.e., $\mathrm{RowSort}(XR) \neq \mathrm{RowSort}(X)$ for a general orthogonal matrix R. Hence, the permutation invariant exploited by VMA does not directly apply to ConjFormer.

  For AloePri, composing the transformed embedding and gate-projection weights yields

  $$\widetilde{W}_e\widetilde{W}_g = \Pi(C_0 + \Delta)Z, \tag{52}$$

  where $C_0 = W_eW_g$ and $\Delta = \alpha_e\mathcal{E}W_g$ captures the perturbation introduced by AloePri. Using VMA in Eq. (49), the vocabulary permutation Π can then be recovered, enabling reconstruction of the private input, although the recovery accuracy is degraded by the perturbation noise.

The key difference between VMA and CROSSWEIGHT is how the unknown right-side permutation Z is handled. Since VMA does not recover Z, it removes Z by row-wise sorting and therefore matches

$$\mathrm{RowSort}(C_0 + \Delta) \quad \text{against} \quad \mathrm{RowSort}(C_0).$$

In contrast, Stage I of CROSSWEIGHT explicitly recovers Z, allowing us to compute

$$\widetilde{W}_e \widetilde{W}_g \widehat{Z}^{-1} = \Pi(C_0 + \Delta)$$

and directly match the aligned rows of $C_0 + \Delta$ against those of C_0.

This distinction is important because row-wise sorting removes not only the nuisance permutation Z, but also the coordinate correspondence within each row. For any two row vectors c_i and c_j,

$$\|\mathrm{sort}(c_i) - \mathrm{sort}(c_j)\|_2 = \min_{P \in \mathcal{P}_m} \|c_i - c_j P\|_2 \leq \|c_i - c_j\|_2. \tag{53}$$

Hence, sorting can only preserve or reduce the separation between different vocabulary-row signatures. AloePri's perturbation Δ therefore more easily changes the nearest-neighbor relation in this sorted space. By recovering Z first, CROSSWEIGHT retains the original coordinate-wise structure and performs matching in the more discriminative aligned space, making vocabulary recovery more robust to the same perturbation.

- **Invariant Attack (IA).** We consider the Gate-IA variant of IA, which exploits a statistical invariant induced by the compatibility between the input transformation and the FFN gate projection. Let $e_v \in \mathbb{R}^{1\times d}$ denote the plaintext embedding of token v. For STIP and CONJFORMER, the corresponding transformed embedding can be written as $\widetilde{e}_v = e_v P_{\mathrm{in}}$. For AloePri, the corresponding row of the transformed embedding table is $\widetilde{e}_{\pi^{-1}(v)} = (e_v + \alpha_e \epsilon_v) P_e$, where ϵ_v is the v-th row of the embedding perturbation $\mathcal{E}$. In all three cases, the input-side and gate-projection transformations form a compatible pair, i.e.,

$$P_{\mathrm{in}} Q_g = I_d$$

for STIP and CONJFORMER, and $P_e Q_g = I_d$ for AloePri. Together with $\widetilde{W}_g = Q_g W_g Z$, this gives

$$\widetilde{e}_v \widetilde{W}_g = e_v W_g Z \tag{54}$$

for STIP and CONJFORMER, whereas for AloePri,

$$\widetilde{e}_{\pi^{-1}(v)} \widetilde{W}_g = (e_v + \alpha_e \epsilon_v) W_g Z. \tag{55}$$

Gate-IA then uses the coordinate mean as a permutation-invariant token signature. For a row vector $x \in \mathbb{R}^{1\times m}$, define

$$s(x) := \mathrm{Avg}(x) = \frac{1}{m} \sum_{j=1}^{m} x_j. \tag{56}$$

Since Z only permutes the coordinates of a vector, it preserves their mean. Therefore, for STIP and CONJFORMER, Eq. (54) yields the exact invariant

$$s\left(\widetilde{e}_v \widetilde{W}_g\right) = s(e_v W_g Z) = s(e_v W_g)\,. \tag{57}$$

Thus, although the gate-projected vector is permuted in the transformed model, its coordinate mean is exactly preserved. For AloePri, the embedding perturbation prevents this exact equality. Taking the mean on both sides of Eq. (55) gives

$$s\left(\widetilde{e}_{\pi^{-1}(v)} \widetilde{W}_g\right) = s(e_v W_g) + \alpha_e s(\epsilon_v W_g)\,. \tag{58}$$

The second term perturbs the signature exploited by Gate-IA, thereby reducing the reliability of signature matching.

Because the plaintext reference model is available under our threat model, the attacker can pre-compute the plaintext signature

$$s_v^{(l)} := \mathrm{Avg}\left(e_v W_g^{(l)}\right) \tag{59}$$

for every vocabulary token v and decoder layer l. For an observed transformed token embedding $\widetilde{e}$, the attacker computes

$$\widetilde{s}^{(l)} := \mathrm{Avg}\left(\widetilde{e}\widetilde{W}_g^{(l)}\right). \tag{60}$$

The signatures from multiple decoder layers are concatenated as

$$\mathbf{s}_v = \left[s_v^{(1)}, \ldots, s_v^{(L)}\right], \qquad \widetilde{\mathbf{s}} = \left[\widetilde{s}^{(1)}, \ldots, \widetilde{s}^{(L)}\right], \tag{61}$$

and the private token is recovered through nearest-neighbor matching, with the same procedure applied to the noisy signatures in AloePri:

$$\widehat{v} = \arg\min_{v\in\mathcal{V}} \|\widetilde{\mathbf{s}} - \mathbf{s}_v\|_2 . \tag{62}$$

- **Embedding Inversion Attack (EIA).** EIA directly attempts to recover private tokens from their observed embeddings by nearest-neighbor matching against the plaintext embedding table. Let $\widetilde{e} \in \mathbb{R}^d$ denote an embedding observed by the server and let $E = \{e_v\}_{v\in\mathcal{V}}$ denote the plaintext token embedding table. The attacker predicts the private token as

$$\widehat{v} = \arg\min_{v\in\mathcal{V}} d(\widetilde{e}, e_v), \tag{63}$$

  where $d(\cdot,\cdot)$ is typically the cosine or Euclidean distance. EIA therefore assumes that the observed embeddings and the plaintext token embeddings lie in the same representation space and have the same dimensionality. This assumption does not hold under the proposed low-rank transformation, where the transformed embedding matrix $\widetilde{W}_e = \Pi W_e \check{P}_e$ produces r-dimensional embeddings, whereas the plaintext embedding table is d-dimensional, with $r < d$. Consequently, direct nearest-neighbor matching between the transformed and plaintext embeddings is infeasible.

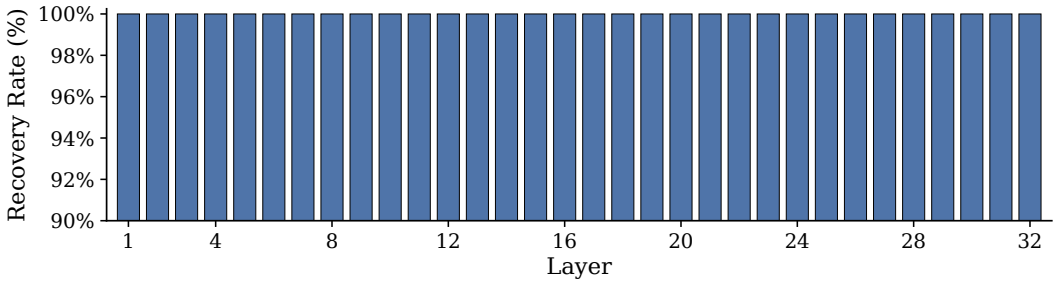
- **Inversion Model Attack (IMA).** IMA considers a stronger training-based attacker that learns an explicit mapping from the transformed embedding space back to the plaintext space. Assuming knowledge of the target transformation mechanism, the attacker iterates over a public training corpus, generates the corresponding transformed embeddings, and uses the resulting plaintext–transformed pairs to train an inversion model.

  Formally, for the proposed low-rank transformation with $\widetilde{e} \in \mathbb{R}^r$ and $e \in \mathbb{R}^d$, the attacker learns

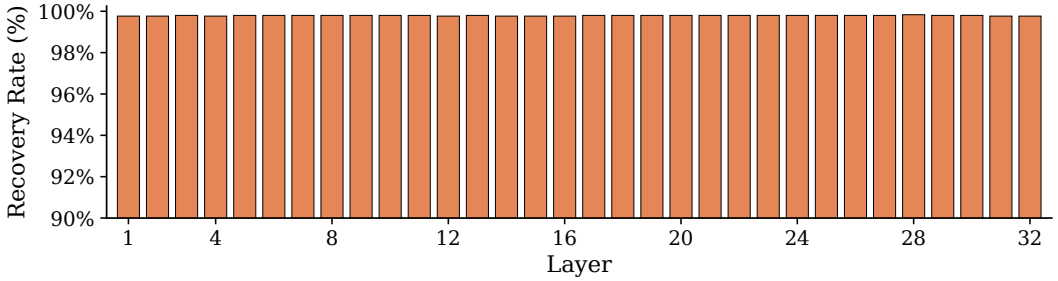
$$f_{\mathrm{inv}} : \mathbb{R}^r \to \mathbb{R}^d, \tag{64}$$

  such that $f_{\mathrm{inv}}(\widetilde{e})$ approximates the corresponding plaintext token embedding. The reconstructed embedding can then be matched against the plaintext embedding table to recover the token. Following AloePri, we instantiate the inversion model using a Qwen2 model with two decoder layers and eight attention heads, trained to map transformed embeddings back to plaintext token embeddings.

Hyperparameters. For the proposed CROSSWEIGHT attack, we separately evaluate MSE and cosine distance as the row-matching metric in Stage I and Stage II and report the best attack performance. For the proposed low-rank transformation mechanism, we set the number of hard-negative candidates to $K = 32$. The weight λ_C is dynamically adjusted such that $\lambda_C\|\nabla L_C\|/\|\nabla L_A\| = 0.2$. For each low-rank transformation pair, we initialize the dimensionality-reduction matrix $P \in \mathbb{R}^{d\times r}$ and the dimensionality-expansion matrix $Q \in \mathbb{R}^{r\times d}$ as $P = B$ and $Q = B^\top$, where $B = [I_r; 0] \in \mathbb{R}^{d\times r}$. For Qwen3-0.6B, the hidden-state dimension is $d = 1024$, and we set $r = 1008$; for Qwen3-1.7B, $d = 2048$, and we set $r = 2016$; for Llama3-8B, $d = 4096$, and we set $r = 4032$. We optimize the trainable parameters using AdamW with a learning rate of 1×10^{-5} and a batch size of 1. We use the `sample-10BT` configuration of FineWeb, from which we select the first 16,000 document records for training. The low-rank transformations are trained for one epoch. All experiments are conducted on a single NVIDIA RTX PRO 6000 GPU. Under this computational budget, we train the proposed low-rank transformations for Qwen3-0.6B, Qwen3-1.7B, and Llama3-8B; experiments on larger models are left for future work.



(a) FFN permutation Z recovery



(b) Input permutation Π recovery

Figure 4: Layer-wise recovery performance of CROSSWEIGHT on Llama3-8B. CROSSWEIGHT achieves exact or near-exact recovery of both the FFN permutation Z and input permutation Π across the evaluated Transformer layers.

Table 4: Utility and privacy comparison on the Llama3-8B model.

Method	**Generation Performance**				**Privacy**		
	PPL ↓	MMLU ↑	PIQA ↑	C-Eval ↑	Attack	TTRSR (%) ↓	CosSim ↓
Plaintext	5.5677	65.11	80.47	52.53	–	–	–
SANTEXT	419.6214	25.25	56.47	25.48	Direct obs.	56.43	0.724
RANTEXT	969.1223	24.70	55.77	24.29	Direct obs.	50.23	0.662
DP-Forward	14165.3447	24.81	48.91	24.81	EIA	61.97	0.741
AloePri	13.5576	27.96	72.58	25.04	CROSSWEIGHT	100	1.000
					VMA	1.60	0.413
					IA	0.00	0.394
AloePri-Noise+	11679.0598	23.97	47.71	25.56	CROSSWEIGHT	68.70	0.779
Proposed	**8.84**	**50.92**	**77.80**	**38.26**	CROSSWEIGHT	3.51	0.406
					VMA	3.49	0.394
					IA	0.00	0.403
					IMA	0.00	0.395

F ADDITIONAL EXPERIMENTAL RESULTS

Figure 4 reports the layer-wise recovery of the FFN permutation Z, together with the corresponding recovery of Π given the recovered Z. The proposed attack achieves consistently high recovery rates across Transformer layers on Llama3-8B, demonstrating its robustness to layer deviation. Figure 5 further presents a qualitative example of text recovery on AloePri. As shown, CROSSWEIGHT exactly reconstructs the source text, whereas existing attack methods fail to recover the original content. This result further demonstrates the effectiveness of CROSSWEIGHT in exploiting the privacy leakage of AloePri.

Table 6 presents an ablation study of the utility and privacy performance of the proposed defense on Qwen3-0.6B under different low-rank dimensions r. As r decreases, privacy protection improves, as evidenced by the lower TTRSR and PIIRSR, while generation performance degrades. This trade-off arises because a smaller r imposes a tighter information bottleneck, resulting in greater information loss during inference. Such information loss makes private inputs more difficult to recover, but

Table 5: Utility and privacy comparison on the Qwen3-0.6B model.

Method	Generation Performance				Privacy		
	PPL ↓	MMLU ↑	PIQA ↑	C-Eval ↑	Attack	TTRSR (%) ↓	CosSim ↓
Plaintext	18.89	47.51	67.57	45.39	–	–	–
SANTEXT	717.81	26.73	52.99	25.26	Direct obs.	62.38	0.673
RANTEXT	688.66	27.54	52.83	28.01	Direct obs.	59.32	0.692
DP-Forward	476671.63	23.82	50.22	23.55	NN	63.08	0.717
AloePri	69.55	29.58	59.15	28.38	CROSSWEIGHT	100	1.000
					VMA	3.21	0.385
					IA	2.30	0.404
AloePri-Noise+	239604.21	23.43	50.87	23.77	CROSSWEIGHT	36.62	0.635
Proposed	**20.54**	**41.85**	**66.05**	**39.38**	CROSSWEIGHT	24.39	0.505
					VMA	0.29	0.383
					IA	0.45	0.395
					IMA	0.00	0.379

Table 6: Ablation study on the low-rank dimension r for Qwen3-0.6B.

Rank r	Generation Performance				Privacy		
	PPL ↓	MMLU ↑	PIQA ↑	C-Eval ↑	Attack	TTRSR (%) ↓	PIIRSR (%) ↓
896	27.90	32.57	62.30	29.49	CROSSWEIGHT	**10.07**	**0.99**
960	23.46	35.52	63.66	34.18		19.72	1.42
1008	**20.54**	**41.85**	**66.05**	**39.38**		24.39	5.06

also reduces the model's generation utility. Nevertheless, compared with AloePri in Table 5, all evaluated rank settings achieve better utility and stronger privacy protection, further demonstrating the effectiveness of the proposed defense mechanism.

Additional results on Llama3-8B and Qwen3-0.6B. Tables 4 and 5 report additional utility–privacy comparisons on Llama3-8B and Qwen3-0.6B, respectively. The results exhibit trends consistent with those observed on Qwen3-1.7B. On Llama3-8B, AloePri remains highly vulnerable to CROSSWEIGHT, with a TTRSR of 100%, while the proposed method reduces the TTRSR to 3.51% and maintains substantially better utility than the DP-based baselines and AloePri-Noise+. In particular, the proposed method achieves a PPL of 8.84, compared with 13.56 for AloePri and $11{,}679.06$ for AloePri-Noise+, while retaining strong performance on MMLU, PIQA, and C-Eval.

A similar trend is observed on Qwen3-0.6B. The proposed method reduces the CROSSWEIGHT TTRSR from 100% on AloePri to 24.39%, while preserving utility close to the plaintext model, with a PPL of 20.54 compared with 18.89 for plaintext. In contrast, increasing the perturbation strength in AloePri-Noise+ reduces the CROSSWEIGHT TTRSR to 36.62% but causes severe utility degradation, with the PPL increasing to $239{,}604.21$. The proposed method also remains robust against VMA, IA, and IMA, with TTRSRs of 0.29%, 0.45%, and 0.00%, respectively. Overall, these results further confirm that the proposed irreversible transformation provides a substantially improved privacy–utility trade-off across different model scales and architectures.

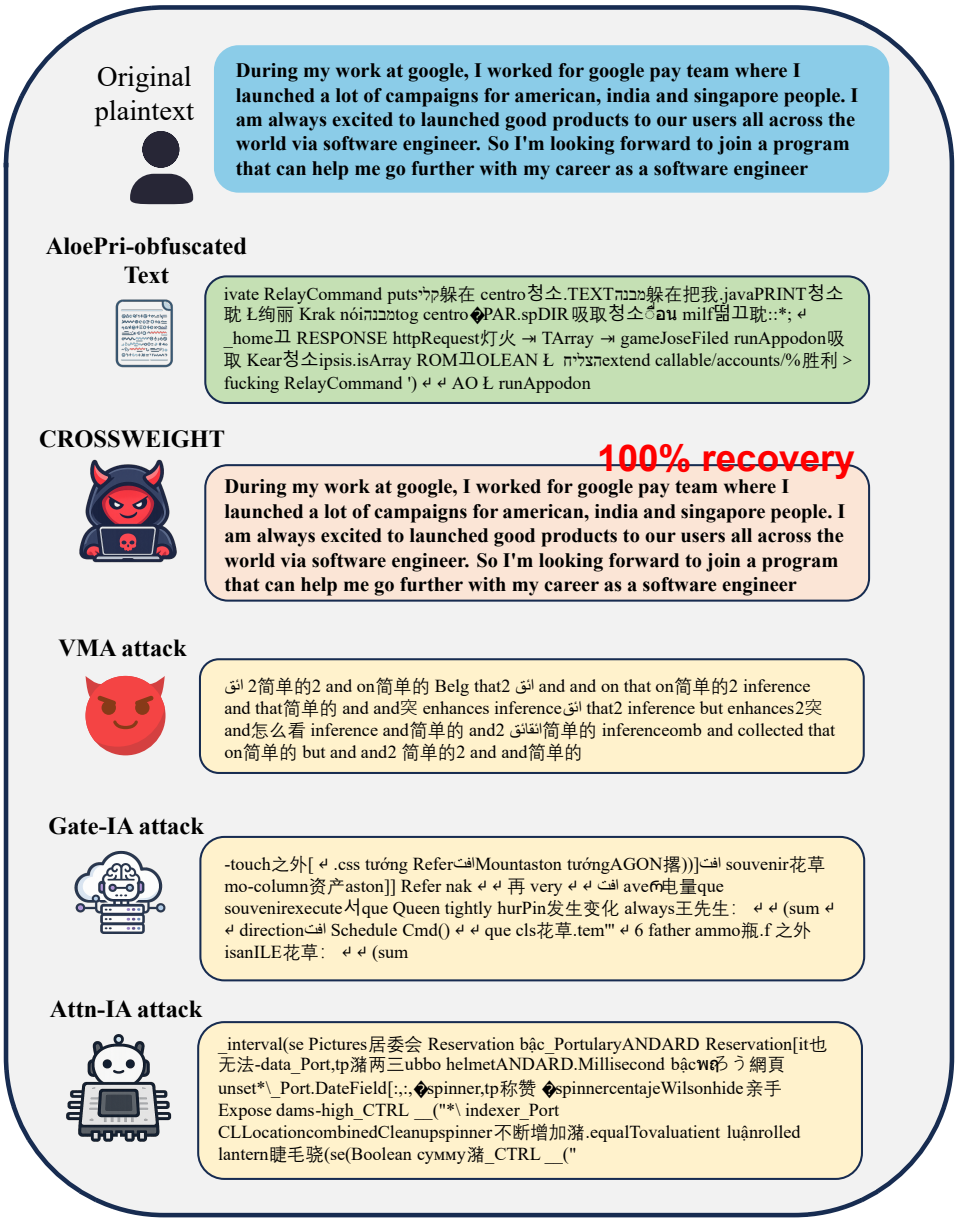


Figure 5: Qualitative example of text recovery attacks on AloePri. While existing attack methods fail to reconstruct the original text, CROSSWEIGHT achieves exact recovery, demonstrating its effectiveness in exploiting the structural leakage of the transformed model.