CENTAUR: Bridging the Impossible Trinity of Privacy, Efficiency, and Performance in Privacy-Preserving Transformer Inference

Jinglong Luo[1,2] **Guanzhong Chen**[1] **Yehong Zhang**[2*] **Shiyu Liu**[5]
Hui Wang[2] **Yue Yu**[2] **Xun Zhou**[1,2*] **Yuan Qi**[3,4] **Zenglin Xu**[2,3,4*]
[1]Harbin Institute of Technology, Shenzhen, [2]Pengcheng Laboratory
[3]Fudan University, [4]Shanghai Academy of AI for Science
[5]Southwestern University of Finance and Economics
`{luojl, zhangyh02}@pcl.ac.cn`, `zhouxun2023@hit.edu.cn`
`zenglin@gmail.com`

Abstract

With the growing deployment of pre-trained models like Transformers on cloud platforms, privacy concerns about model parameters and inference data are intensifying. Existing Privacy-Preserving Transformer Inference (PPTI) frameworks face the "*impossible trinity*" of balancing privacy, efficiency, and performance: Secure Multi-Party Computation (SMPC)-based approaches ensure strong privacy but suffer from high computational overhead and performance losses; Conversely, permutation-based methods achieve near-plaintext efficiency and accuracy but compromise privacy by exposing sensitive model parameters and intermediate results. Bridging this gap with a single approach presents substantial challenges, motivating the introduction of CENTAUR, a groundbreaking PPTI framework that seamlessly integrates random permutations and SMPC to address the "impossible trinity". By designing efficient PPTI algorithms tailored to the structural properties of Transformer models, CENTAUR achieves an unprecedented balance among privacy, efficiency, and performance. Our experiments demonstrate CENTAUR's ability to resist diverse data reconstruction attacks, achieve plaintext-level inference accuracy, and boost inference speed by 5.0∼30.4 times, unlocking new possibilities for secure and efficient AI deployment.

1 Introduction

Transformer-based models (Vaswani et al., 2017; Devlin et al., 2019; Radford et al., 2019), widely deployed in cloud services such as chatbots, virtual assistants, and code generators, have revolutionized many aspects of human activity. However, their cloud-based deployment introduces significant privacy risks. Companies deploying these models and users of the services must upload proprietary model parameters—critical to their competitive edge—along with potentially sensitive inference data, which could include personal information (e.g., identity, investment plans, or health records). These risks not only threaten the competitiveness of companies but also compromise individuals' privacy, raising concerns about whether cloud-based AI models can truly be trusted with sensitive information. Recently, Samsung banned its employees from using external large language model (LLM) services after an internal code leak[1], further underscoring the growing privacy concerns.

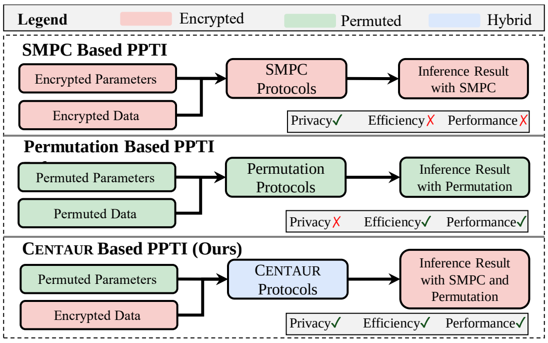

Legend
Encrypted
Permuted
Hybrid
SMPC Based PPTI
Encrypted Parameters
Encrypted Data
SMPC Protocols
Inference Result with SMPC
Privacy✓ Efficiency✗ Performance✗
Permutation Based PPTI
Permuted Parameters
Permuted Data
Permutation Protocols
Inference Result with Permutation
Privacy✗ Efficiency✓ Performance✓
CENTAUR Based PPTI (Ours)
Permuted Parameters
Encrypted Data
CENTAUR Protocols
Inference Result with SMPC and Permutation
Privacy✓ Efficiency✓ Performance✓

Figure 1: Overview of CENTAUR and Other PPTI Frameworks.

Recent works (Hao et al., 2022; Chen et al., 2022; Li et al., 2023; Luo et al., 2024; Yuan et al., 2023) have explored addressing the privacy concerns of model parameters and inference data in Transformer-based inference. However, these approaches often face the *"impossible trinity" of privacy, efficiency, and performance*. For example, SMPC-based privacy-preserving Transformer inference (PPTI) offers strong theoretical privacy guarantees but suffers from significant communication overhead. This inefficiency arises primarily from the *numerous large-scale matrix multiplications and SMPC-unfriendly non-linear operations* inherent in Transformer models. To mitigate these

[*] Corresponding author

[1] https://www.androidauthority.com/samsung-chatgpt-leak-3310307/

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 22751–22770
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

issues, some studies (Li et al., 2023; Luo et al., 2024) have replaced non-linear operations with linear ones, but this substitution results in further performance degradation (see Section 3 for details).

In contrast, permutation-based PPTI (Yuan et al., 2023) achieves efficiency and performance comparable to plaintext inference by conducting plaintext computations on permuted model parameters and inference data. However, to ensure inference correctness, permutation-based PPTI must expose *embedding layer parameters and some original intermediate results*, thereby introducing significant privacy leakage risks (see Section 3 for details).

Existing PPTI frameworks struggle to balance privacy, efficiency, and performance, limiting their practical adoption in real-world applications. To bridge the "impossible trinity" and unlock new possibilities for secure and efficient AI deployment, we propose CENTAUR, a practical PPTI framework that leverages the complementary strengths of multiple privacy-preserving strategies to protect the privacy of both model parameters and inference data (Fig. 1). Specifically:

- **Privacy:** CENTAUR introduces a novel PPTI workflow, ensuring that model parameters, inference data, and intermediate results during inference remain either encrypted or in a randomly permuted state. The security analysis (Section 4.4) and experimental results of data reconstruction attacks (Section 5.2) demonstrate that CENTAUR effectively safeguards the privacy of both model parameters and inference data.
- **Efficiency:** CENTAUR leverages random permutation to transform privacy-preserving multiplications between ciphertexts, which incur high communication overhead, into communication-free operations between plaintexts and ciphertexts, significantly improving the inference efficiency of *linear layers* in PPTI. Additionally, it reduces the communication overhead of *non-linear* operations in PPTI through the design of a series of privacy-preserving algorithms. Experimental results (Section 5.3) show that CENTAUR achieves inference speeds 5.0∼30.4 times faster than existing SMPC-based PPTI frameworks.
- **Performance:** CENTAUR preserves the original model structure and parameters by implementing precise computation of non-linear operations in Transformer models. Experimental results (Section 5.4) demonstrate that CENTAUR achieves performance identical to plaintext inference without the need for retraining or fine-tuning.

2 Preliminaries

2.1 Transformer Models

The Transformer model mainly consists of three components: the *embedding layer*, the *transformer layer*, and the *adaptation layer*. In the embedding layer, the input features of the model are extracted as embeddings. At the transformer layer, the embedded information is processed through a multi-head attention mechanism and passed into the feed-forward neural network to produce a hidden state. In the adaptation layer, the hidden state is ultimately transformed into a vector representation that can be applied to various downstream tasks such as text classification and text prediction.

2.2 Secure Multi-Party Computation

Secure Multi-Party Computation (SMPC) enables a group of untrusted participants to jointly compute a function f without revealing private data. Among the various cryptographic primitives used to implement SMPC, secret sharing (Shamir, 1979; Goldreich et al., 1987) is widely employed in PPTI due to its efficiency. Specifically, 2-out-of-2 secret sharing divides a secret x in the integer ring $\mathbb{Z}_L$ into two random shares $[\![x]\!] = ([x]_0, [x]_1)$, where neither share independently reveals any information about x. The secret can be reconstructed by combining the shares as $x = (([x]_0 + [x]_1) \mod L)$. In two-party SMPC protocols, these shares are distributed among *two non-colluding parties*, who exchange masked intermediate results to perform privacy-preserving computations for various functions. At the end of the process, they each receive shares of the computed results.

2.3 Permutation Matrix

A permutation matrix π is a square matrix containing only 0s and 1s, with exactly one "1" in each row and column. In linear algebra, an $n \times n$ permutation matrix represents a permutation of n elements and has the following key properties:

- Multiplying a matrix by π permutes its rows (if π is on the left) or columns (if π is on the right).
- π is orthogonal, i.e., $\pi\pi^\top = I$.

These properties make permutation matrices useful for privacy-preserving computations in Transformer models, enabling the following operations:

- **Linear Layers:** For a linear layer with parameters (W, B),

$$Y = X\pi(W\pi)^\top + B = XW^\top + B. \quad (1)$$

- **Element-Wise Non-Linear Layers:** For an element-wise non-linear function f_e,

$$f_e(X\pi) = f_e(X)\pi. \quad (2)$$

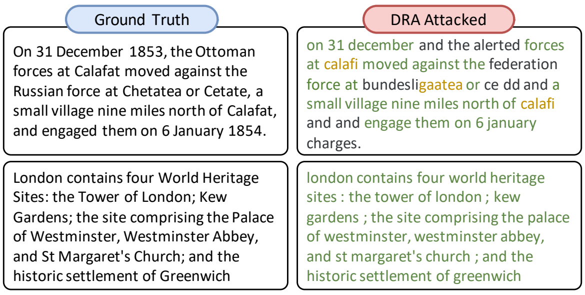
The privacy offered by π increases with its size, making it ideal for large-scale Transformers. Specifically, an $n \times n$ matrix has $n!$ possible permutations. For example, when $n = 1280$, the probability of brute-force recovery of the original matrix is approximately $\frac{1}{1280!} \approx \frac{1}{2^{11372}}$.

3 Impossible Trinity in PPTI

Observation 1: Efficiency and Performance Challenges of SMPC-Based PPTI. SMPC-based PPTI can be formalized as a two-party SMPC protocol between the model developer and the client. In this setup, the shares of model parameters and inference data are used as inputs to the SMPC protocols, enabling privacy-preserving execution of Transformer operations.

This approach ensures privacy for model parameters and inference data but faces severe inefficiencies, primarily from the high communication overhead in large-scale matrix multiplications and non-linear operations within Transformers. For example, running BERT$_{\text{BASE}}$ inference with CrypTen (Knott et al., 2021) in a WAN (200 Mbps bandwidth, 40 ms latency) takes 881 seconds, with 865 seconds spent on transmitting 66 GB of intermediate data.

Efforts to improve the efficiency of SMPC-based PPTI can be classified into two categories: 1) SMPC Protocol Design: Approaches such as (Hao et al., 2022; Zheng et al., 2023; Gupta et al., 2023; Dong et al., 2023; Hou et al., 2023; Ding et al., 2023; Pang et al., 2023; Lu et al., 2023; Luo et al., 2024; Li et al., 2024) focus on developing efficient privacy-preserving algorithms for non-linear operations in Transformers. While these methods preserve model performance, they still incur substantial computation and communication overhead. 2) Model Design: Techniques like (Li et al., 2023; Zeng et al., 2022; Zhang et al., 2023; Liang et al., 2023) modify the model by replacing SMPC-unfriendly non-linear operations to reduce high computational overhead. Although these strategies improve efficiency, they often result in significant performance degradation (see Table 2 for details).



Ground Truth	DRA Attacked
On 31 December 1853, the Ottoman forces at Calafat moved against the Russian force at Chetatea or Cetate, a small village nine miles north of Calafat, and engaged them on 6 January 1854.	on 31 december and the alerted forces at calafi moved against the federation force at bundesligaatea or ce dd and a small village nine miles north of calafi and and engage them on 6 january charges.
London contains four World Heritage Sites: the Tower of London; Kew Gardens; the site comprising the Palace of Westminster, Westminster Abbey, and St Margaret's Church; and the historic settlement of Greenwich	london contains four world heritage sites : the tower of london ; kew gardens ; the site comprising the palace of westminster, westminster abbey, and st margaret's church ; and the historic settlement of greenwich

Figure 2: Two examples of recovering private inference input data through attacks on intermediate results. On the left are the real data, and on the right are the data reconstructed using data reconstruction attacks. Green indicates complete recovery, while orange signifies approximate recovery.

Observation 2: Privacy Leakage Risks in Permutation-Based PPTI. Unlike SMPC-based PPTI, permutation-based PPTI uses permuted model parameters and inference data as input. By leveraging the properties of the permutation matrix, it correctly performs linear layers (matrix multiplication, Eq. (1)) and nonlinear layers (element-wise operations, Eq. (2)), producing permuted inference results. Since the computation is directly performed on the plaintext permuted data, permutation-based PPTI achieves efficiency and performance comparable to plaintext inference. However, it *compromises the privacy of both model parameters and inference data*.

For model parameters, permutation-based PPTI faces the issue of sequence-level permutation vulnerability due to the relatively short length of the inference data sequence. Yuan et al. (2023) suggest performing the permutation in the input feature space[2]. While this method enhances privacy, it requires the model developer to expose the *embedding layer parameters* to the data owner.

Regarding inference data, the orthogonality of the permutation matrix (Eq. (1)) leads to permutation-based PPTI *revealing some original intermediate results*. We have demonstrated that existing data reconstruction methods can effectively recover the private inference data from these raw intermediate results. Fig. 2 illustrates real examples of recovering the original data from the raw intermediate results, and more detailed attack results are provided in Section 5.2.

[2] The feature dimension d is typically large; for example, GPT-2$_{\text{LARGE}}$ has $d = 1280$.

4 CENTAUR

To bridge the "*impossible trinity*" in PPTI, CENTAUR introduces a novel approach that seamlessly integrates random permutations and SMPC. This allows CENTAUR to overcome the limitations of existing methods, achieving a unique balance among privacy, efficiency, and performance. The following sections delve into CENTAUR's design and implementation.

4.1 Framework

CENTAUR focuses on the three-party scenario where the model developer and the cloud platform are separate entities, which is common in real-world model inference service providers (Yuan et al., 2023). Specifically, as shown in Fig. 3, CENTAUR involves three entities: model developer $\mathcal{P}_0$, cloud platform $\mathcal{P}_1$, and client $\mathcal{P}_2$. In this setup, $\mathcal{P}_0$ holds the private model parameters Θ, while $\mathcal{P}_2$ holds the private inference data X.

4.2 Threat Model

CENTAUR adopts the widely used three-party semi-honest model (Wagh et al., 2019; Ryffel et al., 2020; Li et al., 2023; Dong et al., 2023). Specifically, it assumes that the model provider $\mathcal{P}_0$ does not collude with the cloud platform $\mathcal{P}_1$ to obtain the client $\mathcal{P}_2$'s private inference data, and likewise, the cloud platform $\mathcal{P}_1$ does not collude with the client $\mathcal{P}_2$ to access the model provider $\mathcal{P}_0$'s proprietary model parameters.

In contrast to two-party PPTI protocols (Hao et al., 2022; Pang et al., 2023; Lu et al., 2023), which require the data owner to act as one of the computing parties and frequently communicate with the model provider during the entire inference process—often relying on homomorphic encryption or oblivious transfer to generate correlated randomness—CENTAUR takes a different approach. The data owner, who typically has limited computing and communication capabilities, only plays the role of a dealer, responsible for generating the correlated randomness required for PPTI execution, such as Beaver triples (Beaver, 1992) for multiplication and random permutation matrices to accelerate computation.

The computation and communication intensive tasks are delegated to the semi-honest cloud platform and the model developer, both of which are assumed to have ample resources. CENTAUR consists of two main phases: Initialization and Privacy-Preserving Inference, detailed as follows.

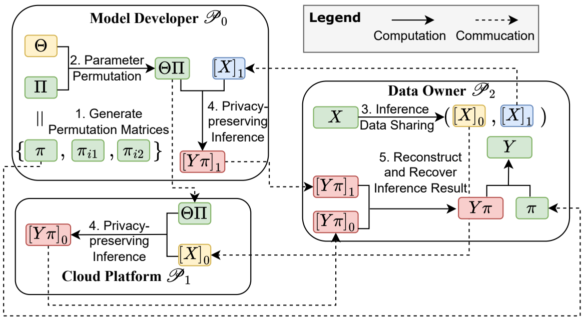

Model Developer $\mathscr{P}_0$
Legend
Computation
Commucation
Θ
Π
2. Parameter Permutation
$\Theta\Pi$
$[X]_1$
1. Generate Permutation Matrices
$\{\pi, \pi_{i1}, \pi_{i2}\}$
4. Privacy-preserving Inference
$[Y\pi]_1$
Data Owner $\mathscr{P}_2$
X
3. Inference Data Sharing
$([X]_0, [X]_1)$
Y
5. Reconstruct and Recover Inference Result
$[Y\pi]_1$
$[Y\pi]_0$
$Y\pi$
π
$\Theta\Pi$
$[Y\pi]_0$
4. Privacy-preserving Inference
$[X]_0$
Cloud Platform $\mathscr{P}_1$

Figure 3: High-level Workflow of CENTAUR.

Initialization. The model developer $\mathcal{P}_0$ generates a set of random permutation matrices, $\Pi = \{\pi \in \mathbb{R}^{d\times d}, \pi_1 \in \mathbb{R}^{n\times n}, \pi_2 \in \mathbb{R}^{k\times k}\}$, where n denotes the input length, d represents the feature dimension, and k corresponds to the intermediate dimension in the feed-forward neural network. These matrices are designed to permute the model parameters according to their respective dimensions. Among them, the permutation matrix π is shared with $\mathcal{P}_2$. Subsequently, $\mathcal{P}_0$ applies the appropriate permutation matrix from Π to permute the model parameters Θ, resulting in the permuted parameters Θ', which are then sent to $\mathcal{P}_1$.

Privacy-Preserving Inference. The client $\mathcal{P}_2$ locally generates shares of the inference data $X \to ([X]_0, [X]_1)$ and sends $[X]_j$ to the respective parties $\mathcal{P}_j$ for $j \in \{0, 1\}$. Each $\mathcal{P}_j$ then takes Θ' and $[X]_j$ as input, and jointly executes the privacy-preserving inference process according to the workflow shown in Fig. 4, resulting in the shares of the permuted inference result $[Y\pi]_j$. Subsequently, each $\mathcal{P}_j$ sends $[Y\pi]_j$ to the client $\mathcal{P}_2$. Upon receiving $[Y\pi]_j$, $\mathcal{P}_2$ reconstructs the permuted inference result $Y\pi = [Y\pi]_0 + [Y\pi]_1$, and restores the final inference result using π: $Y = Y\pi\pi^\top$.

4.3 Implementation

As described in Section 2.1, the Transformer model consists of the Transformer layers, the embedding layer, and the adaptation layer. We now outline how CENTAUR enhances privacy-preserving computation in each of these layers.

4.3.1 Transformer Layers

Linear Layer. CENTAUR optimizes the efficiency of linear layers by converting costly privacy-preserving matrix multiplications between random shares (denoted as Π_{MatMul}) into communication-free privacy-preserving multiplications between

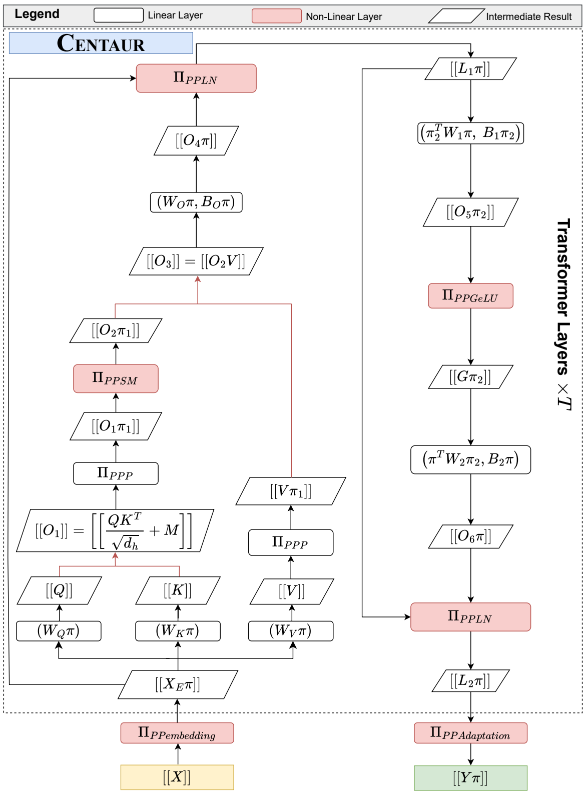


Figure 4: Implementation of CENTAUR-based PPTI. Red lines and boxes indicate that there is a communication overhead for the computation of this step. Black lines indicate completion of the calculation for that step without communication overhead.

plaintexts and random shares (denoted as Π_{ScalMul}). This is achieved by separately using random permutation for model parameters and secret-sharing for inference data to ensure privacy.

As shown in Fig. 4, the linear layer parameters consist of $W_Q, W_K, W_V, (W_O, B_O)$ in the attention mechanism and $(W_1, B_1), (W_2, B_2)$ in the feed-forward neural network for a single Transformer block. During the initialization phase, these parameters are permuted by the model developer $\mathcal{P}_0$. When data, in the form of secret shares, passes through these linear layers, the computation is performed using the communication-free plaintext-shares privacy-preserving multiplication protocol Π_{ScalMul}. The shares of the computation results are then output as follows:

$$
\begin{aligned}
[\![Q]\!] &= \Pi_{\text{ScalMul}}(W_Q\pi, [\![X_E\pi]\!]),^{3} \\
[\![K]\!] &= \Pi_{\text{ScalMul}}(W_K\pi, [\![X_E\pi]\!]), \\
[\![V]\!] &= \Pi_{\text{ScalMul}}(W_V\pi, [\![X_E\pi]\!]), \\
[\![O_4\pi]\!] &= \Pi_{\text{ScalMul}}(W_O\pi, [\![O_3]\!]) + B_O\pi, \\
[\![O_5\pi_2]\!] &= \Pi_{\text{ScalMul}}(\pi_2^\top W_1\pi, [\![L_1\pi]\!]) + B_1\pi_2, \\
[\![O_6\pi]\!] &= \Pi_{\text{ScalMul}}(\pi^\top W_2\pi_2, [\![G\pi_2]\!]) + B_2\pi.
\end{aligned}
\tag{3}
$$

To ensure the correctness and security of the inference results, CENTAUR requires a limited number of privacy-preserving matrix multiplications between shares in the attention mechanism. The detailed computation process is as follows:

$$
\begin{aligned}
[\![O_1]\!] &= \Pi_{\text{MatMul}}([\![Q]\!], [\![K]\!])/\sqrt{d_h} + [\![M]\!], \\
[\![O_3]\!] &= \Pi_{\text{MatMul}}([\![O_2\pi_1]\!], [\![V\pi_1]\!]).
\end{aligned}
\tag{4}
$$

Non-linear Layers. CENTAUR enhances the efficiency of nonlinear layers by converting secret shares into a randomly permuted state, enabling plaintext computations for element-wise nonlinear operations on the permuted data.

For any nonlinear operation with permuted input $X\pi$, which has been secret-shared between $\mathcal{P}_0$ and $\mathcal{P}_1$, the process proceeds as follows:

- The model developer $\mathcal{P}_0$ sends the share $[X\pi]_0$ to the cloud platform $\mathcal{P}_1$, enabling it to convert the input from the secret-sharing state $[\![X\pi]\!]$ to the permuted state $X\pi$.
- $\mathcal{P}_1$ performs the nonlinear computation using $X\pi$ and obtains the permuted output $Y\pi$.
- $\mathcal{P}_1$ generates shares $[\![Y\pi]\!]$ of $Y\pi$ and sends $[Y\pi]_0$ back to $\mathcal{P}_0$.

This process requires two rounds of communication to transmit the shares of both the input and the output. Based on this, CENTAUR supports Privacy-Preserving Softmax (Π_{PPSM}), Privacy-Preserving GeLU (Π_{PPGeLU}), and Privacy-Preserving LayerNorm (Π_{PPLN}) for computing nonlinear layers in Transformers. Detailed construction algorithms are provided in Appendix A.

It is important to note that transitioning the input from the secret-sharing state $[\![X\pi]\!]$ to the permuted state $X\pi$ requires the input shares to be in the permuted state. However, this condition is not always met in the PPTI process. For example, the shares of O_1 are initially not in the permuted state because the permutation matrix π is canceled out during Π_{MatMul} (Eq. (4)). To address this, CENTAUR introduces a Privacy-Preserving Permutation (Π_{PPP}) protocol. By invoking privacy-preserving matrix multiplication, Π_{PPP} converts the shares of any input $[\![X]\!]$ into $[\![X\pi]\!]$. The detailed process is outlined in Algorithm 6.

[3] We omit the bias here for concise presentation. For the case that there is additional bias parameters B in producing Q, K, and V, the model developer can secretly share B to cloud platform and add it to the output of Π_{ScalMul} using Π_{Add}.

		BERT$_{\text{LARGE}}$ on the QNLI dataset					GPT-2$_{\text{LARGE}}$ on the Wikitext-103 dataset				
Attacks	Methods	O_1	O_4	O_5	O_6	Avg	O_1	O_4	O_5	O_6	Avg
SIP	W/O	66.14 ± 1.38	78.64 ± 0.28	95.57 ± 0.06	96.00 ± 0.05	84.09	69.64 ± 0.68	92.91 ± 0.17	93.69 ± 0.11	94.31 ± 0.21	87.64
	W(Ours)	10.72 ± 2.01	$\underline{2.03 \pm 0.89}$	$\underline{0.00 \pm 0.00}$	$\underline{2.71 \pm 1.86}$	$\underline{3.86}$	$\underline{6.10 \pm 4.67}$	12.90 ± 0.64	0.58 ± 0.21	$\underline{2.00 \pm 1.29}$	5.40
	Rand	$\underline{5.08 \pm 0.04}$	6.82 ± 0.02	0.17 ± 0.06	3.58 ± 0.21	3.91	14.65 ± 0.90	$\underline{2.69 \pm 0.05}$	$\underline{0.00 \pm 0.00}$	3.38 ± 0.04	$\underline{5.18}$
EIA	W/O	100.00 ± 0.00	36.49 ± 1.13	80.97 ± 0.71	19.5 ± 0.50	59.24	96.70 ± 0.02	99.97 ± 0.04	100.00 ± 0.00	67.30 ± 0.01	90.99
	W(Ours)	1.37 ± 0.12	$\underline{5.94 \pm 0.43}$	2.89 ± 0.13	$\underline{0.12 \pm 0.07}$	2.58	1.36 ± 0.10	11.90 ± 0.37	7.91 ± 0.23	4.40 ± 0.33	6.39
	Rand	$\underline{0.14 \pm 0.00}$	7.22 ± 0.17	$\underline{0.34 \pm 0.11}$	0.85 ± 0.03	$\underline{2.13}$	$\underline{0.30 \pm 0.02}$	$\underline{8.27 \pm 0.02}$	$\underline{2.54 \pm 0.06}$	$\underline{4.29 \pm 0.04}$	$\underline{3.85}$
BRE	W/O	56.64 ± 1.06	14.85 ± 0.55	74.50 ± 0.75	7.80 ± 0.11	38.45	56.64 ± 1.06	99.99 ± 0.01	99.99 ± 0.00	45.26 ± 0.58	75.47
	W(Ours)	0.21 ± 0.02	0.45 ± 0.03	0.52 ± 0.39	$\underline{0.52 \pm 0.39}$	0.43	0.21 ± 0.02	1.33 ± 0.07	$\underline{0.03 \pm 0.01}$	$\underline{0.07 \pm 0.02}$	0.41
	Rand	$\underline{0.07 \pm 0.02}$	$\underline{0.25 \pm 0.20}$	$\underline{0.09 \pm 0.01}$	0.58 ± 0.02	$\underline{0.25}$	$\underline{0.07 \pm 0.02}$	$\underline{0.20 \pm 0.00}$	0.08 ± 0.00	0.10 ± 0.01	$\underline{0.11}$

Table 1: The degree of privacy leakage (ROUGE-L F1 Score (%)) on the permuted intermediate results "O_1, O_4, O_5, O_6" using three data reconstruction attack methods. "W/O" represents the original data, "W" represents the permuted state, and "Rand" represents random input. The results denote the attack targets and are averaged over three different random seeds.

4.3.2 Embedding Layer & Adaptation Layer.

The Embedding and Adaptation layers involve both linear and nonlinear operations, enabling dual acceleration of efficiency within CENTAUR. Specifically, the Embedding layer includes matrix multiplication and LayerNorm operations, allowing for Privacy-Preserving Embedding ($\Pi_{\text{PPEmbedding}}$) via the invocation of Π_{ScalMul} and Π_{PPLN}. The construction of the Privacy-Preserving Adaptation ($\Pi_{\text{PPAdaptation}}$) layer, which adapts to different downstream tasks such as classification or prediction, varies across Transformer models. However, it can be uniformly implemented by using CENTAUR's privacy-preserving algorithms. Detailed constructions for PPEmbedding and PPAdaptation are provided in Algorithm 4 and Algorithm 5.

4.4 Theoretical Analysis

CENTAUR can leverage the properties of permutation matrices to ensure the confidentiality of model parameters. By applying the widely used simulation-based paradigm (Lindell, 2017) from SMPC, we can demonstrate that intermediate results under secret-sharing can guarantee the confidentiality of user inference data. Additionally, using distance correlation theory (SZÉKELY et al., 2007), privacy protection of intermediate results under random permutation can be analyzed. We leave the detailed security analyses in Appendix B since the theoretical framework of the SMPC and random permutation mechanism, which is usually the focus of the security community, is not over-explored by CENTAUR. We will focus on substantiating the *empirical security* of CENTAUR through rich and complex attack experiments in Section 5.2.

5 Experiments

We conducted experiments to address three key questions regarding CENTAUR: **Q1 (Privacy)**: Does the intermediate result in CENTAUR, stored in a randomly permuted state, withstand various rigorous adversarial attacks? **Q2 (Efficiency)**: Can CENTAUR improve the inference speed of PPTI? **Q3 (Performance)**: Does CENTAUR maintain the model's performance during PPTI execution?

5.1 Experimental Setup

Implementation. We perform CENTAUR on CrypTen, a privacy-preserving machine learning framework based on SMPC. Our experiments were conducted on three servers, each equipped with an A100 GPU. To assess efficiency under varying conditions, we simulated different network settings using Linux Traffic Control. For the Local Area Network (LAN), the bandwidth was set to 3 Gbps with a round-trip delay of 0.8 ms, while for the Wide Area Network (WAN), the bandwidth was 100 Mbps with an 80 ms delay.

Baselines. CENTAUR is compared with several state-of-the-art PPTI frameworks: MPCFormer (Li et al., 2023), PUMA (Dong et al., 2023), and SecFormer (Luo et al., 2024). MPCFormer improves PPTI efficiency by replacing Softmax and GeLU with SMPC-friendly quadratics. PUMA optimizes PPTI efficiency with enhanced SMPC protocols for nonlinear operations, while SecFormer also replaces Softmax with SMPC-friendly quadratics and refines protocols for nonlinear layers.

Models and Datasets. To ensure fairness, we selected the BERT and GPT-2 models, which are widely used in baseline evaluations, as benchmark models for our experimental assessment. For the BERT model, we selected five datasets from the GLUE benchmark (Wang et al., 2019) (RTE,

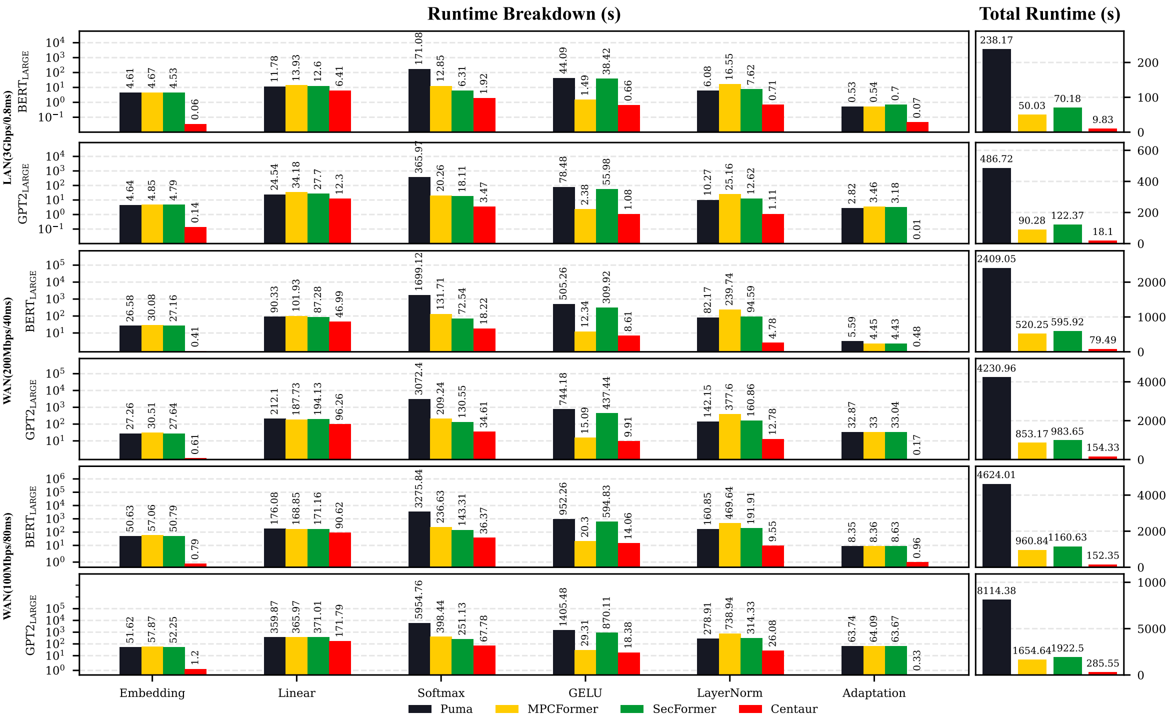


Figure 5: Time breakdown for each operations (left) and the entire PPTI process (right) of the tested frameworks. The results are the average of ten runs.

CoLA, STS-B, MRPC, QNLI) for natural language understanding (NLU) tasks. For GPT-2, we employed two datasets from the Wikitext collection (Merity et al., 2017) (Wikitext-103 and Wikitext-2) for natural language generation (NLG) tasks. Furthermore, as CENTAUR performs the computation of nonlinear functions in Transformer models under permutation, it can theoretically be easily extended to other types of Transformer models, such as LLaMA, while maintaining a better balance between privacy, efficiency, and performance. These details will be further outlined in Appendix E.

5.2 Empirical Security

To answer **Q1**, we conduct a series of rigorous adversarial experiments. Specifically, we first employ the three most advanced Data Reconstruction Attack (DRA) methods to attack the permuted intermediate results in an attempt to retrieve private inference data from users without recovering the permutation matrix. We also perform pattern-based and heuristic-based methods to recover the permutation matrix from the permuted intermediate results. The attack setup and more results of the attack experiments are presented in Appendix C.

Attack Methods. We evaluate three mainstream DRA methods targeting the intermediate outputs of Transformer models: (1) SIP (Chen et al., 2024), a learning-based approach that trains an inversion model on the auxiliary dataset to reconstruct the original sentence from any intermediate output derived from the private dataset; (2) Embedding Inversion Attack (EIA) (Song and Raghunathan, 2020), an optimization-based approach that generates a dummy input and iteratively optimizes it (through relaxed optimization within the discrete vocabulary space) to match the observed intermediate outputs; and (3) BRE (Chen et al., 2024), an optimization-based approach that constructs dummy inputs but performs optimization within the continuous embedding space.

Attack Targets. In CENTAUR, as outlined in Section 4.3, intermediate results such as $O_1\pi_1$, $O_4\pi$, $O_5\pi_2$, and $O_6\pi$ are stored in permuted form on cloud platform P_1. To validate the privacy protection capabilities of CENTAUR, we conduct DRA experiments on these permuted results. For comparison, we also set up two control experiments: one with the original intermediate results (O_1, O_4, O_5, O_6), and another with random matrices of equivalent dimensions. The focus of our experiments is on the first Transformer block, where privacy leakage is most likely to occur.

Evaluation Metrics. We use ROUGE-L (Rouge,

2004) F1 score as the evaluation metric for the attack experiments. ROUGE-L F1 assesses similarity based on the longest common subsequence, strictly following the order and tokens. By analyzing the ROUGE-L F1 values, we can understand the extent to which the original inference data can be reconstructed from the intermediate results. The ROUGE-L F1 score ranges from 0 to 1, with lower values indicating a lower recovery rate.

Evaluation Results. The experimental results in Table 1 demonstrate that on both BERT$_{\text{LARGE}}$ and GPT-2$_{\text{LARGE}}$, the average ROUGE-L F1 values for data recovery by the three attack methods using CENTAUR's permuted intermediate results are comparable to those obtained with random inputs. This further confirms that CENTAUR effectively preserves the privacy of inference data. Specifically, for BERT$_{\text{LARGE}}$, the three attack methods recover only 3.86%, 2.58%, and 0.43% of the data's average ROUGE-L F1 values on the QNLI classification task dataset. In contrast, the recovery rate significantly increases when original intermediate results are used for the attacks. For instance, on GPT-2$_{\text{LARGE}}$, the average ROUGE-L F1 value for data recovery using EIA reaches as high as 90.99%, with 100% data recovery achieved on $O_1 = QK^{\top}$. This indicates that the privacy-preserving mechanism of PPTI (Yuan et al., 2023) based on random permutation completely fails once the original intermediate results are exposed.

5.3 Efficiency Comparison

To address **Q2**, we analyze the inference time and communication overhead of CENTAUR performing PPTI and compare it with current state-of-the-art frameworks. The key results are presented in Fig. 5, with more details provided in Appendix D. In two network settings—LAN (3Gbps, 0.8ms) and WAN (100Mbps, 80ms)—CENTAUR significantly outperforms other PPTI frameworks. For BERT$_{\text{LARGE}}$, CENTAUR is 5.1∼24.2 times faster in a LAN environment and 6.3∼30.4 times faster in WAN. For GPT-2$_{\text{LARGE}}$, CENTAUR is 5.0∼26.9 times faster in LAN and 5.8∼28.4 times faster in WAN. These efficiency improvements are attributed to CENTAUR's dual optimization of both the linear and non-linear layers within PPTI.

Linear Layers. CENTAUR speeds up inference in linear layers by 1.8∼2.2 times for BERT$_{\text{LARGE}}$ and 2.0∼2.8 times for GPT-2$_{\text{LARGE}}$ compared to other PPTI frameworks. This is due to CENTAUR's use of randomly permuted model parameters and secret-shared inference data, allowing most linear computations to be performed with the communication-free private matrix multiplication protocol Π_{ScalMul}.

Non-Linear Layers. In the non-linear layers, CENTAUR achieves significant speed-ups. For Softmax and GeLU, CENTAUR outperforms the SMPC-based framework PUMA by two orders of magnitude. For BERT$_{\text{LARGE}}$, CENTAUR is 3.2∼93.3 times faster in Softmax, 1.4∼66.8 times faster in GeLU, and 8.6∼50.1 times faster in LayerNorm. For GPT-2$_{\text{LARGE}}$, the speed-ups are 3.7∼105.5, 1.5∼76.5, and 9.3∼29.5 times, respectively. These improvements are attributed to the privacy-preserving non-linear algorithms proposed in CENTAUR, which significantly reduce the communication overhead of non-linear computations in PPTI by converting the secret-share state to a random permutation state.

Embedding & Adaptation Layers. The embedding and adaptation layers, which involve both linear and non-linear operations, benefit from CENTAUR's dual optimization. For BERT$_{\text{LARGE}}$, CENTAUR's inference speed in the embedding layer is 364.1∼377.8 times faster, while for GPT-2$_{\text{LARGE}}$, the speedup ranges from 67.1∼82.8 times. In the adaptation layer, CENTAUR accelerates BERT$_{\text{LARGE}}$ by 7.6∼11.6 times and GPT-2$_{\text{LARGE}}$ by 193.7∼290.9 times.

5.4 Performance Comparison

To answer **Q3**, we validate the performance of CENTAUR and show the results in Table 2. As can be seen, both the BERT series models with an encoder structure and the GPT series models with a decoder structure achieve the same performance when using CENTAUR for PPTI as inference in plaintext. This indicates that CENTAUR does not compromise the performance of the plaintext models while protecting the model parameters and inference data. This is because CENTAUR does not make any adjustments to the structure of the plaintext Transformer models during the PPTI process. Consequently, CENTAUR can be combined with any existing Transformer architecture model to achieve PPTI with performance equivalent to plaintext inference.

6 Discussion

CENTAUR bridges the "impossible trinity" of privacy, efficiency, and performance in privacy-

	QNLI (108k)	CoLA (8.5k)	STS-B (5.7k)	MRPC (3.5k)	RTE (2.5k)	Avg.	Wikitext-2 (45k)	Wikitext-103 (1800k)	Avg.
	BERT$_{\text{BASE}}$(↑)						GPT-2$_{\text{BASE}}$(↓)		
Plain-text	91.7	57.8	89.1	90.3	69.7	79.7	20.3	24.3	22.3
PUMA	91.7	57.8	89.1	90.3	69.7	79.7	20.3	24.3	22.3
● MPCFormer$_{w/o}$	69.8	0.0	36.1	81.2	52.7	48.0	420.9	520.0	470.5
● MPCFormer	90.6	52.6	80.3	88.7	64.9	75.4	431.8	522.3	477.1
○ SecFormer$_{w/o}$	89.3	57.0	86.2	83.8	63.2	75.9	75.4	131.0	103.2
○ SecFormer	91.2	57.1	87.4	89.2	69.0	78.8	75.3	130.9	103.1
CENTAUR (Ours)	91.7	57.8	89.1	90.3	69.7	79.7	20.3	24.3	22.3
	BERT$_{\text{LARGE}}$(↑)						GPT-2$_{\text{LARGE}}$(↓)		
Plain-text	92.4	61.7	90.2	90.6	75.5	82.1	14.4	16.0	15.2
PUMA	92.4	61.7	90.2	90.6	75.5	82.1	14.4	16.0	15.2
● MPCFormer$_{w/o}$	49.5	0.0	0.0	81.2	52.7	36.7	94.4	396.2	245.3
● MPCFormer	87.8	0.0	52.1	81.4	59.2	56.1	94.5	402.5	248.5
○ SecFormer$_{w/o}$	90.8	60.8	89.0	87.6	69.7	79.6	91.8	143.1	117.5
○ SecFormer	92.0	61.3	89.2	88.7	72.6	80.8	91.5	140.6	119.1
CENTAUR (Ours)	92.4	61.7	90.2	90.6	75.5	82.1	14.4	16.0	15.2

Table 2: Performance comparison of BERT and GPT-2 models. Underlined numbers indicate the best results. Marker ○ refer to approximating GeLU with Quad. Marker ● refer to approximating GeLU and Softmax with Quad and 2Quad, respectively. "w/o" indicates no re-training or knowledge distillation

preserving transformer inference (PPTI) by leveraging the complementary strengths of SMPC and random permutation strategies. Comprehensive experiments demonstrate that CENTAUR significantly improves the efficiency of PPTI while providing a practical level of privacy, without sacrificing model performance. This enables CENTAUR to be readily integrated into existing Transformer-based model-as-a-service (MaaS) platforms to support privacy-preserving inference.

Moreover, CENTAUR does not yet incorporate other techniques designed to improve the computational and memory efficiency of Transformer inference, such as quantization and KV-cache, which could further enhance overall PPTI efficiency. These techniques are orthogonal to CENTAUR, and integrating them poses new challenges. For instance, KV-cache involves operations that are inherently incompatible with SMPC—such as similarity computation, top-k selection, and token aggregation—which would require additional considerations to enable privacy-preserving KV-cache. We leave the exploration of these directions for future work, aiming to further enhance the privacy and efficiency of PPTI by combining CENTAUR with such optimizations.

7 Conclusion

This paper introduces CENTAUR, an efficient PPTI framework that employs tailored privacy-preserving mechanisms for both model parameters and inference data. By seamlessly integrating these techniques with customized algorithms, CENTAUR strikes an optimal balance in the privacy-efficiency-performance trade-off, often referred to as the "*impossibility triangle*", unlocking new possibilities for the secure deployment of language models.

8 Limitations

CENTAUR adopts a privacy-preserving mechanism based on random permutation, which means that it cannot directly achieve theoretical security. CENTAUR does not overemphasize the theoretical security frameworks focused on the security domain but instead supports its claimed empirical security through extensive and complex attack experiments. In practical applications, privacy and usability are often incompatible. Particularly in the era of large models based on Transformer architectures, the rapid growth in model size has made traditional provable security techniques, such as SMPC and homomorphic encryption, impractical due to their high communication and computational costs. Therefore, we believe exploring practical privacy-preserving mechanisms for large models is of significant importance. Among various unverifiable security methods, the privacy-preserving capabilities of random permutation are positively correlated with the scale of the protected entity, making it especially suitable for large models with high-dimensional Transformer architectures. CENTAUR achieves a better balance between privacy and usability by combining random permutation with other provable security techniques. At the same time, we believe that the practical attack analyses on intermediate results in language models performed in CENTAUR hold equal importance to purely theoretical frameworks and require evaluation by the NLP community.

9 Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (No. 62206139, No. 62472125, No. 72495122), the Natural Science Foundation of Guangdong Province, China (No. 2025A1515011258), and Shenzhen Sustained Support for Colleges & Universities Program (No. GXWD20231128102922001), the Major Key Project of PCL (PCL2023A09).

References

Anton O. Bassin and Maxim Buzdalov. 2020. The $(1 + (\lambda, \lambda))$ genetic algorithm for permutations. In *GECCO '20: Genetic and Evolutionary Computation Conference, Companion Volume*, pages 1669–1677. ACM.

Donald Beaver. 1992. Efficient multiparty protocols using circuit randomization. In *Advances in Cryptology—CRYPTO'91: Proceedings 11*, pages 420–432. Springer.

Ran Canetti. 2001. Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science*, pages 136–145. IEEE.

Guanzhong Chen, Zhenghan Qin, Mingxin Yang, Yajie Zhou, Tao Fan, Tianyu Du, and Zenglin Xu. 2024. Unveiling the vulnerability of private fine-tuning in split-based frameworks for large language models: A bidirectionally enhanced attack. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, CCS '24, page 2904–2918.

Tianyu Chen, Hangbo Bao, Shaohan Huang, Li Dong, Binxing Jiao, Daxin Jiang, Haoyi Zhou, Jianxin Li, and Furu Wei. 2022. THE-X: Privacy-preserving transformer inference with homomorphic encryption. In *Findings of the Association for Computational Linguistics*, pages 3510–3520.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186.

Yuanchao Ding, Hua Guo, Yewei Guan, Weixin Liu, Jiarong Huo, Zhenyu Guan, and Xiyong Zhang. 2023. East: Efficient and accurate secure transformer framework for inference. *arXiv preprint arXiv:2308.09923*.

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

Oded Goldreich, Silvio Micali, and Avi Wigderson. 1987. How to play any mental game or A completeness theorem for protocols with honest majority. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 218–229. ACM.

Kanav Gupta, Neha Jawalkar, Ananta Mukherjee, Nishanth Chandran, Divya Gupta, Ashish Panwar, and Rahul Sharma. 2023. SIGMA: Secure GPT inference with function secret sharing. *Cryptology ePrint Archive, Paper 2023/1269*.

Meng Hao, Hongwei Li, Hanxiao Chen, Pengzhi Xing, Guowen Xu, and Tianwei Zhang. 2022. Iron: Private inference on transformers. *Advances in Neural Information Processing Systems*, 35:15718–15731.

Xiaoyang Hou, Jian Liu, Jingyu Li, Yuhan Li, Wen jie Lu, Cheng Hong, and Kui Ren. 2023. CipherGPT: Secure two-party GPT inference. *Cryptology ePrint Archive, Paper 2023/1147*.

Brian Knott, Shobha Venkataraman, Awni Hannun, Shubho Sengupta, Mark Ibrahim, and Laurens van der Maaten. 2021. CrypTen: Secure multi-party computation meets machine learning. *Advances in Neural Information Processing Systems*, 34:4961–4973.

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

Zhengyi Li, Kang Yang, Jin Tan, Wen-jie Lu, Haoqi Wu, Xiao Wang, Yu Yu, Derun Zhao, Yancheng Zheng, Minyi Guo, et al. 2024. Nimbus: Secure and efficient two-party inference for transformers. *arXiv preprint arXiv:2411.15707*.

Zi Liang, Pinghui Wang, Ruofei Zhang, Nuo Xu, and Shuo Zhang. 2023. MERGE: Fast private text generation. *arXiv preprint arXiv:2305.15769*.

Yehuda Lindell. 2017. How to simulate it – A tutorial on the simulation proof technique. *Tutorials on the Foundations of Cryptography*, pages 277–346.

Wen-jie Lu, Zhicong Huang, Zhen Gu, Jingyu Li, Jian Liu, Cheng Hong, Kui Ren, Tao Wei, and WenGuang Chen. 2023. Bumblebee: Secure two-party inference framework for large transformers. *Cryptology ePrint Archive*.

Jinglong Luo, Yehong Zhang, Jiaqi Zhang, Xin Mu, Hui Wang, Yue Yu, and Zenglin Xu. 2024. Secformer: Towards fast and accurate privacy-preserving inference for large language models. *arXiv preprint arXiv:2401.00793*.

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

Stanley RM Oliveira and Osmar R Zaiane. 2004. Privacy-preserving clustering by object similarity-based representation and dimensionality reduction transformation. In *Proceedings of the ICDM Workshop on Privacy and Security Aspects of Data Mining*, pages 40–46.

Qi Pang, Jinhao Zhu, Helen Möllering, Wenting Zheng, and Thomas Schneider. 2023. BOLT: Privacy-preserving, accurate and efficient inference for transformers. *Cryptology ePrint Archive, Paper 2023/1893*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Lin CY Rouge. 2004. A package for automatic evaluation of summaries. In *Proceedings of Workshop on Text Summarization of ACL, Spain*, volume 5.

Théo Ryffel, Pierre Tholoniat, David Pointcheval, and Francis Bach. 2020. AriaNN: Low-interaction privacy-preserving deep learning via function secret sharing. *Proc. on Privacy Enhancing Technologies*, 2022(1):291–316.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083.

Adi Shamir. 1979. How to share a secret. *Communications of the ACM*, 22(11):612–613.

Congzheng Song and Ananth Raghunathan. 2020. Information leakage in embedding models. In *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security*, pages 377–390.

GÁBOR J SZÉKELY, MARIA L RIZZO, and NAIL K BAKIROV. 2007. Measuring and testing dependence by correlation of distances. *The Annals of Statistics*, 35(6):2769–2794.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*.

Sameer Wagh, Divya Gupta, and Nishanth Chandran. 2019. SecureNN: 3-Party secure computation for neural network training. *Proceedings on Privacy Enhancing Technologies*, pages 26–49.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR*.

Yining Wang, Yu-Xiang Wang, and Aarti Singh. 2018. A theoretical analysis of noisy sparse subspace clustering on dimensionality-reduced data. *IEEE Transactions on Information Theory*, 65(2):685–706.

Mu Yuan, Lan Zhang, and Xiang-Yang Li. 2023. Secure transformer inference. *arXiv preprint arXiv:2312.00025*.

Wenxuan Zeng, Meng Li, Wenjie Xiong, Wenjie Lu, Jin Tan, Runsheng Wang, and Ru Huang. 2022. MPCViT: Searching for MPC-friendly vision transformer with heterogeneous attention. *arXiv preprint arXiv:2211.13955*.

Yuke Zhang, Dake Chen, Souvik Kundu, Chenghao Li, and Peter A Beerel. 2023. SAL-ViT: Towards latency efficient private inference on ViT using selective attention search with a learnable softmax approximation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5116–5125.

Fei Zheng, Chaochao Chen, Xiaolin Zheng, and Mingjie Zhu. 2022. Towards secure and practical machine learning via secret sharing and random permutation. *Knowledge-Based Systems*, 245:108609.

Mengxin Zheng, Qian Lou, and Lei Jiang. 2023. Primer: Fast private transformer inference on encrypted data. *arXiv preprint arXiv:2303.13679*.

Appendices

The appendices in this paper are organized as follows.

- Appendix A presents the privacy-preserving algorithms designed in CENTAUR.
- Appendix B offers a detailed theoretical security analysis of the CENTAUR framework.
- Appendix C offers a detailed empirical security analysis of the CENTAUR framework.
- Appendix D presents comparative analyses of CENTAUR's communication volume and inference time for BERT$_{\text{BASE}}$ and GPT-2$_{\text{BASE}}$.
- Appendix E provides a comprehensive analysis of CENTAUR framework, further supported by experimental results on the LLaMA-7B model.
- Finally, Appendix F outlines the hyperparameters employed in the performance experiments.

A Privacy-preserving Algorithms in CENTAUR

In this section, we present the construction of privacy-preserving algorithms within CENTAUR. Specifically, this includes Privacy-Preserving Softmax (Π_{PPSM}), Privacy-Preserving GeLU (Π_{PPGeLU}), Privacy-Preserving LayerNorm (Π_{PPLN}), Privacy-preserving permutation (Π_{PPP}), Privacy-Preserving Embedding ($\Pi_{\text{PPEmbedding}}$), and Privacy-Preserving Adaptation ($\Pi_{\text{PPAdaptation}}$). We illustrate the construction of $\Pi_{\text{PPAdaptation}}$ using the BERT series model as an example. In the BERT model, the Adaptation layer consists of a pooling layer composed of a linear layer (W_P, B_P) and the activation function Tanh, followed by a linear layer with parameters (W_C, B_C).

Algorithm 1: Privacy-preserving Softmax (Π_{PPSM})

```
Input: For $j \in \{0,1\}$, $\mathcal{P}_j$ holds $[X\pi]_j$.
Output: For $j \in \{0,1\}$, $\mathcal{P}_j$ holds
        $[Y\pi]_j = [\text{Softmax}(X)\pi]_j$.
The model developer $\mathcal{P}_0$ transmits $[X\pi]_0$ to $\mathcal{P}_1$
$\mathcal{P}_1$ reconstructs $X\pi$ and calculates
    $Y\pi = \text{Softmax}(X\pi) = \text{Softmax}(X)\pi$
$\mathcal{P}_1$ generates shares of $Y\pi$ and sends $[Y\pi]_0$ to $\mathcal{P}_0$
```

Algorithm 2: Privacy-preserving GeLU (Π_{PPGeLU})

```
Input: For $j \in \{0,1\}$, $\mathcal{P}_j$ holds $[X\pi_2]_j$.
Output: For $j \in \{0,1\}$, $\mathcal{P}_j$ holds
        $[Y\pi_2]_j = [\text{GeLU}(X)\pi_2]_j$.
The model developer $\mathcal{P}_0$ sends $[X\pi_2]_0$ to $\mathcal{P}_1$
$\mathcal{P}_1$ reconstructs $X\pi_2$ and calculates
    $Y\pi_2 = \text{GeLU}(X\pi_2)$
$\mathcal{P}_1$ generates shares of $Y\pi_2$ and sends $[Y\pi_2]_0$ to $\mathcal{P}_0$
```

Algorithm 3: Privacy-preserving LayerNorm (Π_{PPLN})

```
Input: For $j \in \{0,1\}$, $\mathcal{P}_j$ holds $[X\pi]_j$.
Output: For $j \in \{0,1\}$, $\mathcal{P}_j$ holds
        $[Y\pi]_j = [\text{LayerNorm}(X)\pi]_j$.
The model developer $\mathcal{P}_0$ transmits $[X\pi]_0$ to $\mathcal{P}_1$
$\mathcal{P}_1$ reconstructs $X\pi$ and calculates
    $Y\pi = \text{LayerNorm}(X\pi, \gamma\pi, \beta\pi)$
$\mathcal{P}_1$ generates shares of $Y\pi$ and sends $[Y\pi]_0$ to $\mathcal{P}_0$
```

Algorithm 4: Privacy-preserving Embedding ($\Pi_{\text{Embedding}}$)

```
Input: For $j \in \{0,1\}$, $\mathcal{P}_j$ holds $[X]_j$.
Output: For $j \in \{0,1\}$, $\mathcal{P}_j$ holds $[X_E\pi]_j$.
$\mathcal{P}_0$ and $\mathcal{P}_1$ jointly calculate
    $[\![X_M\pi]\!] = \Pi_{\text{ScalMul}}([\![input]\!], W_E\pi)$
$\mathcal{P}_0$ and $\mathcal{P}_1$ jointly calculate
    $[\![X_E\pi]\!] = \Pi_{\text{PPLN}}([\![X_M\pi]\!])$
```

Algorithm 5: Privacy-preserving Adaptation ($\Pi_{\text{Adaptation}}$)

```
Input: For $j \in \{0,1\}$, $\mathcal{P}_j$ holds $[X\pi]_j$.
Output: For $j \in \{0,1\}$, $\mathcal{P}_j$ holds $[Y\pi]_j$.
$\mathcal{P}_0$ and $\mathcal{P}_1$ jointly calculate
    $[\![X_P\pi]\!] = \Pi_{\text{ScalMul}}([\![input]\!], W_P\pi)$
The model developer $\mathcal{P}_0$ sends $[X\pi]_0$ to $\mathcal{P}_1$
$\mathcal{P}_1$ reconstructs $X\pi$ and calculates
    $T\pi = \text{Tanh}(X\pi) = \text{Tanh}(X)\pi$
$\mathcal{P}_1$ generates shares of $T\pi$ and sends $[T\pi]_0$ to $\mathcal{P}_0$
$\mathcal{P}_0$ and $\mathcal{P}_1$ jointly calculate
    $[\![Y]\!] = \Pi_{\text{ScalMul}}([\![T\pi]\!], W_c)$
```

Algorithm 6: Privacy-preserving permutation (Π_{PPP})

```
Input: For $j \in \{0,1\}$, $\mathcal{P}_j$ holds $[X]_j$.
Output: For $j \in \{0,1\}$, $\mathcal{P}_j$ holds $[X\pi]_j$.
$\mathcal{P}_2$ generates a random permutation $\pi \in \mathbb{R}^{d \times d}$
$\mathcal{P}_2$ generates the shares $([\pi]_0, [\pi]_1)$ and sends $[\pi]_j$ to $\mathcal{P}_j$
$\mathcal{P}_0$ and $\mathcal{P}_1$ jointly calculate the permuated share $[\![X\pi]\!] = \Pi_{\text{MatMul}}([\![X]\!], [\![\pi]\!])$.
```

B Theoretical Analysis

In this section, we theoretically demonstrate that CENTAUR can protect the confidentiality of both model parameters held by the model developer and user inference data. Specifically, we first leverage the properties of permutation matrices and the Transformer model structure to show how CENTAUR ensures the confidentiality of model parameters. Next, by applying the widely-used simulation-based paradigm from secure multi-party computation (SMPC), we illustrate how intermediate results in a secret-sharing state can safeguard the confidentiality of user inference data. Furthermore, we analyze the privacy-preserving capabilities of intermediate results under random permutation using distance correlation theory.

B.1 Privacy of Model Parameters

In CENTAUR, the permutation matrices $\Pi = \{\pi, \pi_1, \pi_2\}$ are randomly generated locally by the model developer $\mathcal{P}_0$ during the initialization phase. Subsequently, $\mathcal{P}_0$ sends the permutation matrix π to the client $\mathcal{P}_2$ and the permuted model parameters to the cloud platform $\mathcal{P}_1$. During the privacy-preserving inference phase, although $\mathcal{P}_1$ receives the permuted parameters in the linear layers and LayerNorm layers $\{W_E\pi, W_Q\pi, W_K\pi, W_V\pi, (W_O\pi, B_O\pi), (\pi_2 W_1 \pi, B_1\pi), (\pi W_2\pi_2, B_2\pi), (\gamma_1\pi, \beta_1\pi), (\gamma_2\pi, \beta_2\pi)\}$, it lacks information about the permutation matrices $\{\pi \in \mathbb{R}^{d\times d}, \pi_2 \in \mathbb{R}^{k\times k}\}$. This prevents $\mathcal{P}_1$ from directly obtaining the original parameters. Based on the properties of permutation matrices, the probability that $\mathcal{P}_1$ can derive the original parameters $\{W_E, W_Q, W_K, W_V, (W_O, B_O), (\gamma_1, \beta_1), (\gamma_2, \beta_2), B_2\}$ from the permuted ones is $\frac{1}{d!}$. The probability of retrieving the parameters $\{W_1, W_2\}$ is $\frac{1}{d!k!}$ and B_1 is $\frac{1}{k!}$.

Also, during both the initialization and privacy-preserving inference phases, the client $\mathcal{P}_2$ can only obtain the permutation matrix π and the permuted inference results, thus preventing any access to information about the model parameters.

B.2 Privacy of Inference Data

Unlike model parameters, inference data in CENTAUR is split into random shares. We prove that CENTAUR can ensure that during PPTI, neither the model developer $\mathcal{P}_0$ nor the cloud platform $\mathcal{P}_1$ can obtain any meaningful information about the inference data. Firstly, we prove through simulation that the intermediate results in the random shares state in CENTAUR do not leak the privacy of the inference data. Then, we demonstrate through distance correlation theory and various attack experiments to verify that the permuted intermediate results do not leak the privacy of the inference data.

Intermediate Results in the Secret-Sharing State. CENTAUR follows the semi-honest (also known as honest-but-curious) assumption, similar to (Li et al., 2023; Dong et al., 2023; Luo et al., 2024). Under this assumption, the security of CENTAUR can be formally proven in the simulation paradigm, particularly against a static semi-honest adversary (denoted as $\mathcal{A}$). Specifically, the simulation paradigm divides the process into two distinct worlds: the real world and the ideal world. In the real world, the server executes the protocol in the presence of a semi-honest adversary $\mathcal{A}$. In contrast, in the ideal world, the server transmits the input information to a trusted dealer who executes the protocol correctly. The security of the CENTAUR framework requires that the protocol executed with intermediate results in a randomly shared state produces distributions in the real world and the ideal world that are indistinguishable for any semi-honest adversary $\mathcal{A}$.

Theorem 1 *The protocols executed in* CENTAUR, *using intermediate results in a randomly shared state as input, satisfies the following criteria:*

- ***Correctness:*** *For a model F_Θ with parameters Θ and inference data X, the output of the client at the end of the protocol is the correct inference result $F_\Theta(X)$.*
- ***Security:*** *For any corrupted computing server $\mathcal{S}_j$ with $j \in \{0,1\}$, there exists a probabilistic polynomial-time simulator $Sim_{\mathcal{S}_j}$ such that the adversary $\mathcal{A}$ cannot distinguish between $View_{\mathcal{S}_j}^{\Pi_P}$ (i.e., the view of $\mathcal{S}_j$ during the execution of Π_P) and $Sim_{\mathcal{S}_j}$.*

		BERT$_{\text{LARGE}}$ on the MRPC dataset					GPT-2$_{\text{LARGE}}$ on the Wikitext-2 dataset				
Attacks	Methods	O_1	O_4	O_5	O_6	Avg	O_1	O_4	O_5	O_6	Avg
SIP	W/O	70.94 ± 0.17	85.40 ± 0.38	97.61 ± 0.08	97.89 ± 0.08	87.96	65.38 ± 0.14	93.59 ± 0.04	93.07 ± 0.13	94.68 ± 0.05	86.68
	W(Ours)	10.96 ± 1.24	$\underline{1.68 \pm 0.29}$	$\underline{2.36 \pm 1.67}$	4.96 ± 0.67	4.99	4.64 ± 0.91	11.58 ± 0.47	0.48 ± 0.29	2.68 ± 2.71	4.85
	Rand	$\underline{5.72 \pm 0.05}$	6.09 ± 0.04	3.79 ± 2.68	$\underline{4.14 \pm 0.19}$	$\underline{4.94}$	$\underline{0.09 \pm 0.01}$	$\underline{1.20 \pm 0.02}$	$\underline{0.00 \pm 0.00}$	$\underline{1.46 \pm 0.01}$	$\underline{0.69}$
EIA	W/O	100.00 ± 0.00	34.25 ± 0.62	78.41 ± 0.50	19.31 ± 0.78	57.99	96.17 ± 0.05	100.00 ± 0.00	99.99 ± 0.01	65.04 ± 2.97	90.30
	W(Ours)	1.60 ± 0.40	$\underline{5.65 \pm 0.47}$	3.41 ± 0.85	$\underline{0.25 \pm 0.21}$	2.73	1.46 ± 0.17	12.49 ± 0.25	8.67 ± 0.20	4.89 ± 0.77	6.88
	Rand	$\underline{0.13 \pm 0.01}$	6.57 ± 0.08	$\underline{0.28 \pm 0.01}$	0.77 ± 0.03	$\underline{1.94}$	$\underline{0.76 \pm 0.80}$	$\underline{9.69 \pm 0.73}$	$\underline{2.13 \pm 0.78}$	$\underline{4.11 \pm 0.36}$	$\underline{4.17}$
BRE	W/O	51.89 ± 1.26	73.30 ± 0.43	70.86 ± 0.37	11.34 ± 2.40	51.85	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	40.50 ± 0.36	85.13
	W(Ours)	$\underline{0.07 \pm 0.01}$	2.77 ± 0.11	1.08 ± 0.20	0.91 ± 0.36	1.21	0.26 ± 0.14	2.14 ± 0.20	$\underline{0.04 \pm 0.01}$	$\underline{0.07 \pm 0.01}$	0.63
	Rand	0.18 ± 0.01	$\underline{1.94 \pm 0.08}$	$\underline{0.68 \pm 0.03}$	$\underline{0.54 \pm 0.05}$	$\underline{0.84}$	$\underline{0.17 \pm 0.06}$	$\underline{0.28 \pm 0.07}$	0.06 ± 0.02	0.09 ± 0.02	$\underline{0.15}$

Table 3: Attack performance (RougeL-F%) on BERT$_{\text{LARGE}}$ and GPT-2$_{\text{LARGE}}$. The MRPC dataset is used for BERT and the Wikitext-2 dataset is used for GPT-2. “W/O” represents the original data without permutation; “W” represents the permuted state; “Rand” represents random input. Results are the average of three different random seeds.

We provide the proof of Theorem 1 through the following analyses. According to Fig. 4 and Eqs. (3)-(4), the linear layers in a Transformer model only involve privacy-preserving operations Π_{PPP} which is essentially a Π_{MatMul}, Π_{ScalMul}, Π_{MatMul}, and Π_{Add}. Since these basic operations Π_{ScalMul}, Π_{MatMul}, and Π_{Add} have been proven to satisfy Theorem 1, we can directly prove that CENTAUR satisfies Theorem 1 for these linear layers using the universally composable security theorem established in (Canetti, 2001).

Intermediate Results in the Randomly Permuted State. In CENTAUR, to perform non-linear operations such as Π_{PPSM}, Π_{PPGeLU}, and Π_{PPLN}, a conversion from a random sharing state to a random permutation state is required. During this process, the model developer $\mathcal{P}_0$ needs to send $[X\pi]_0$ to the cloud platform $\mathcal{P}_1$ for the reconstruction of $X\pi$, resulting in the intermediate results being in a random permutation state.

We demonstrate both theoretically and experimentally that intermediate results in a random permutation state do not leak the privacy of inference data. Specifically, from a theoretical standpoint, we employ distance correlation theory (SZÉKELY et al., 2007) to prove that the privacy leakage caused by intermediate results in a randomly permuted state is less than that of one-dimensional reduction, which has already been proven to possess privacy-preserving capabilities in practical applications (Wang et al., 2018; Oliveira and Zaiane, 2004). According to (Zheng et al., 2022), for any vector $o \in \mathbb{R}^{1\times d}$, the following inequality holds:

$$\begin{aligned} &\mathop{\mathbb{E}}_{\pi, W_A \in \mathbb{Z}^{d\times d}}[\text{Discorr}(o, oW_A\pi)] \\ \le &\mathop{\mathbb{E}}_{W_B \in \mathbb{Z}^{d\times 1}}[\text{Discorr}(o, oW_B)], \end{aligned} \tag{5}$$

where Discorr denotes a distance correlation function. This inequality implies that the distance correlation of the vector o after passing through a linear layer with parameter W_A, followed by a permutation π, is less than or equal to the distance correlation after passing through a linear layer W_B that compresses it to a 1-dimensional output. According to Fig. 4, all shares pass through at least one linear layer before being converted to a permuted state in CENTAUR. Therefore, it can be proven that the intermediate results in the permuted state in CENTAUR satisfy Eq. (5).

C Empirical Security Analysis

In this section, we demonstrate that the distributed secure inference based on the permutation of intermediate results provides *empirical privacy security*. Specifically, in the scenario we consider, even for a reasonably strong attacker, the difficulty of successfully launching an attack is extremely high. To illustrate this, we assume an *overly idealized* adversary, who has full white-box access to all parts of the model segment held by the model developer. This assumption is unrealistic in practical application scenarios. To comprehensively evaluate privacy, we further categorize the adversary into two types: those who launch attacks *with and without cracking the permutation matrix*. It is important to note that, to date, no existing work has successfully compromised permuted Transformer intermediate results. We are the first to conduct a thorough analysis of the privacy and security of permutation-based Transformer inference.

Attack Setup. We evaluate the privacy protection capabilities of CENTAUR by conducting a series of data reconstruction attack (DRA) experiments, with and without the adversary attempting to crack the permutation matrix (secret key). Consider an *overly idealized attack scenario* where the adver-

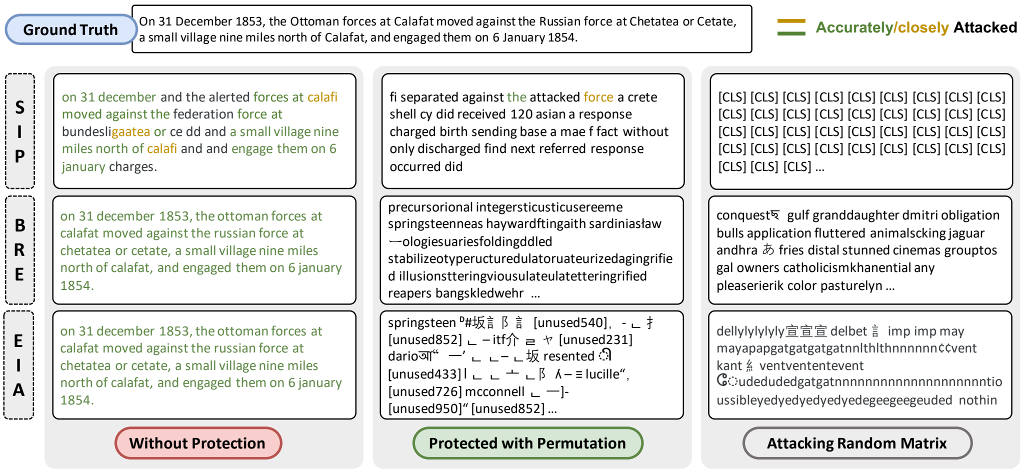

Ground Truth
On 31 December 1853, the Ottoman forces at Calafat moved against the Russian force at Chetatea or Cetate, a small village nine miles north of Calafat, and engaged them on 6 January 1854.
Accurately/closely Attacked
SIP
BRE
EIA
on 31 december and the alerted forces at calafi moved against the federation force at bundesligaatea or ce dd and a small village nine miles north of calafi and and engage them on 6 january charges.
fi separated against the attacked force a crete shell cy did received 120 asian a response charged birth sending base a mae f fact without only discharged find next referred response occurred did
[CLS] ...
on 31 december 1853, the ottoman forces at calafat moved against the russian force at chetatea or cetate, a small village nine miles north of calafat, and engaged them on 6 january 1854.
precursorional integersticusticusereeme springsteenneas haywardftingaith sardiniasław —ologiesuariesfoldingddled stabilizeotyperucturedulatoruateurizedagingrifie d illusionstteringviousulateulatetteringrified reapers bangskledwehr ...
conquestक्ष gulf granddaughter dmitri obligation bulls application fluttered animalscking jaguar andhra あ fries distal stunned cinemas grouptos gal owners catholicismkhanential any pleaserierik color pasturelyn ...
on 31 december 1853, the ottoman forces at calafat moved against the russian force at chetatea or cetate, a small village nine miles north of calafat, and engaged them on 6 january 1854.
springsteen ᴰ#坂言 阝 言 [unused540], - ㄴ 扌 [unused852] ㄴ – itf介 ⊇ ヤ [unused231] darioआ“ —’ ㄴ ㄴ– ㄴ坂 resented ঀ [unused433] l ㄴ ㄴ ㅗ ㄴ阝 ʎ– ≡ lucille“, [unused726] mcconnell ㄴ —]- [unused950]“ [unused852] ...
dellylylylyly宣宣宣 delbet 言 imp imp may mayapapgatgatgatgatnnlthlthnnnnnn¢¢vent kant 糸 ventventententevent ઉ◌udedudedgatgatnnnnnnnnnnnnnnnnnntio ussibleyedyedyedyedegeegeegeuded nothin
Without Protection
Protected with Permutation
Attacking Random Matrix

Figure 6: An example of recovering private inference input data through O_1.

sary has unrestricted query access to key intermediate components of the model. An adversary can launch attacks at any nonlinear intermediate layer and recover the inference data's privacy using only the intermediate results from that layer. Additionally, we assume this powerful adversary has access to an auxiliary dataset that may or may not resemble the target private dataset. We use a batch size of 4 and evaluate the average attack performance on 20 batches. To ensure the stability of the experimental results, each set of experiments was conducted with three different random seeds. The CNN-DailyMail News Text Summarization dataset (See et al., 2017), which is entirely distinct from the target private datasets, was selected as the auxiliary dataset to simulate a realistic attack scenario.

C.1 Attack without Cracking the Permutation Matrix

For an attacker who does not attempt to crack the permutation matrix, the inability to determine whether the target intermediate results have been permuted leads them to employ traditional DRA strategies designed for the intermediate results of Transformers. In Section 5.2, we have already provided experimental results for three state-of-the-art data reconstruction attack methods tailored for this scenario. Here, we present the attack setup and implementation details for the three adopted DRA methods, along with additional results and specific examples from the attack experiments discussed earlier.

Implementation Details. For SIP, we employ a simple GRU model as the Inversion Model, with a hidden size of 256 and a dropout rate of 0.1, and train it for 20 epochs on the CNN Daily-Mail News dataset. Given that the last two dimensions of O_1 correspond to variable-length sequences, we truncate these sequences to a fixed length (512 in our experiments) before inputting them into the Inversion Model for training. For EIA, we use the Gumbel Softmax approximation to construct a distribution matrix over the vocabulary, which is then fed into the model. We optimize the intermediate outputs using Euclidean distance as the loss function. Since the attack focuses on intermediate results from the first layer, we do not need to apply the mapping strategy to shallow layers as described in (Song and Raghunathan, 2020). For BRE, we directly construct an embedding, bypassing the embedding layer, and input it into the language model, optimizing based on cosine similarity. We conduct 6000 epochs of optimization for BRE and 2400 epochs for EIA, with both methods using AdamW with a learning rate of 0.1 as the optimizer.

More Attack Result. We also report the outcomes of attacks on the MRPC dataset using the BERT$_{\text{LARGE}}$ model and on the Wikitext-2 dataset using the GPT-2$_{\text{LARGE}}$ model. Specifically, for the BERT$_{\text{LARGE}}$ model, the average ROUGE-L F1 scores for data recovery across three different attack methods on the MRPC classification task dataset are a mere 4.99%, 2.73%, and 1.21%, respectively. These results are comparable to the ROUGE-L F1 scores obtained when attacking random inputs. In contrast, attacks on plaintext intermediate results yield significantly higher recovery rates. Notably, the average ROUGE-L F1 score for data recovered using SIP from plaintext intermediate results reaches as high as 87.96%. A similar pattern is observed with the GPT-2$_{\text{LARGE}}$ model during prediction tasks. On the Wikitext-2 dataset, the average ROUGE-L F1 scores for data recovery from randomly permuted intermediate results are 4.58%, 6.88%, and 0.63%, which are again comparable to the recovery rates from random inputs. However, when targeting plaintext intermediate results, the average ROUGE-L F1 scores for data recovery using the three attack methods are significantly higher, with the EIA method recovering over 90.3% of the private data.

Attack Examples. We provide additional practical attack examples targeting $O_1 = QK^T$. These examples clearly demonstrate that directly attacking the plaintext O_1 can effectively recover private inference data, indicating that permutation-based PPTI presents a significant privacy leakage risk. In contrast, attacking obfuscated intermediate results or random inputs only produces meaningless garbled output. This demonstrates that the privacy protection provided by CENTAUR can effectively resist current DRA attacks.

Analysis. For the considered data reconstruction attacks, to launch an attack based on observations in the intermediate space N, the attacker must obtain an inverse mapping f^{-1} to map the results back to the vocabulary space V. In this context, we investigate the correlation between the proportion of shuffled features in the intermediate results and the effectiveness of f^{-1}. The results, after fitting and smoothing, are presented in Fig. 7. It is evident that a small amount of feature displacement (20%) can significantly reduce the effectiveness of f^{-1}. In practice, for the permutation matrix generated by np.permutation, when the hidden size of the large language model (LLM) exceeds 768, the proportion of non-shuffled elements is less than 0.13%, effectively achieving near-complete feature reordering. This leads to the complete disruption of f^{-1}. Thus, although the intermediate representations of the Transformer are sparse, in a scenario where almost all features are randomly reordered, *any direct attack method that does not consider cracking the permutation matrix is impractical.*

C.2 Attack by Cracking the Permutation Matrix

Furthermore, we consider a more advanced adversary, who is aware that the intermediate result being attacked has been permuted and attempts to launch an attack by cracking the permutation matrix. We emphasize that permuting the intermediate results of a Transformer is *difficult to crack in practice*, which stems from:

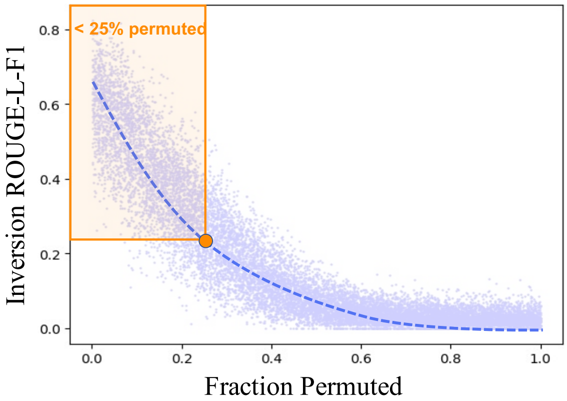


Figure 7: The correlation between the proportion of shuffled features and the effectiveness (measured by the ROUGE-L F1 score) of the inversion attack f^{-1}, showing that reconstructing the raw text requires 75%+ of the features to remain in place.

- **Huge secret key space**: A typical Transformer model has a large dimensionality for its intermediate representations. For instance, the dimensionality is often 768 (and it is even over a thousand for GPT-2 and Llama). The key space reaches 768!. Even for a computer with a computing power of 10^{18} FLOPS, it is impossible to solve the problem within a reasonable time frame.
- **Noisiness of intermediate representations**: Usually, the cracking of substitution ciphers is carried out directly in the vocabulary space V. However, in the context of Centaur, the target model $f : V \rightarrow N$ maps the original sentences to the intermediate space N. For the attacker under consideration, the cracking process occurs in the space N. For an attacker aiming to reconstruct the original sentence data from a target, *the function f is noisy*. Due to the stacking of attention mechanisms, the intermediate activations of the same token vary across different contexts and also differ from the initial embedding of that token. That is to say, the randomness here comes from the context during inference.

Due to the adversary's limited attack view caused by the perturbation, the large key space, and the challenging inversion curve shown in Fig. 7, cracking the permutation matrix proves to be difficult in practice. In the following, we attempt two cracking methods, namely pattern-based and searching-based approaches, both of which fail to successfully break the permutation matrix.

C.2.1 Crack by Pattern Identification: Difficult

For permutation-based encryption in the feature space, a key issue is whether there are identifiable patterns across the feature dimensions that could be exploited by the attacker to launch a cracking attack. We note that, due to operations such as LayerNorm performed by the Transformer on the feature dimensions, it is difficult to attempt cracking by simply identifying patterns in the different features, as *their distributions are too similar to be distinguished*.

Table 4: The average global Jensen-Shannon (JS) divergence across the feature dimensions of the intermediate results generated during inference on different models and datasets, with all values being less than 0.1, indicates that the distribution differences across the feature dimensions are minimal.

Model	PIQA	WikiText	MRPC	QNLI
BERT-large	0.0748	0.0618	0.0605	0.0612
GPT2-large	0.0555	0.0441	0.0462	0.0470

Distribution Similarity Test We calculated the global Jensen-Shannon (JS) divergence (ranging from 0 to 1, where 1 indicates a clear distinction between distributions) among all feature dimensions of the intermediate activations on BERT, GPT-2, and three datasets. It can be observed from Table 4 that all the global JS divergences are less than 0.1. The differences in the distributions of different features are extremely small. Moreover, considering that the intermediate dimension is quite large (>= 768), it is very difficult in practice to recover the permutation matrix by observing the distributions of these features.

Classifier-based Test We also attempted to use RNN and Linear as classifiers to model the distribution characteristics of different feature dimensions. However, even after careful tuning, such classifiers failed to fit successfully during the training process.

C.2.2 Crack by Heuristic Searching: Difficult

Cracking strategies that use heuristic signals such as frequency as search guides are indeed efficient in traditional substitution cipher scenarios. However, in the scenario considered by Centaur, *the presence of noise makes it difficult for attackers to find effective and accurate heuristic signals*.

Take the frequency-based attack as an example. Different from the monoalphabetic substitution cipher, the substitution space (768!) and the vocabulary space (>10000) are much larger than the alphabet. Moreover, the "substitution" occurs in the intermediate results rather than the original vocabulary, even if an attacker might obtain the intermediate representation of a known token, they still cannot directly solve for the permutation matrix as in a known-plaintext attack (KPA), because there is a random perturbation between the intermediate representation they possess and the one they observe.

We conducted experiments to further verify the difficulty of heuristic search attacks. We consider both genetic algorithm (Bassin and Buzdalov, 2020) and gradient-based continuous approximation approaches for searching the permutation matrix. We tried various heuristic schemes to guide the cracking process, including:

- Frequency-based. The attacker can use the clustering of intermediate results (since permutation does not affect clustering based on metrics such as cosine-similarity) to count token frequencies. After identifying high-frequency tokens, the attacker can crack the permutation matrix by comparing the intermediate representations of high-frequency tokens before and after permutation. In the experiment, we assumed that the attacker had completely determined the identities of the top-5 and top-1 high-frequency permuted intermediate results. We attempted to use the cosine similarity between the original intermediate results of these five tokens (sampled by the attacker from the auxiliary dataset) and the observed values as a heuristic signal.

- Scoring-model-based. An ML model can be used to model the relationship between the "degree of disorder" and the degree of restoration mentioned in Fig. 7. This model takes the permuted intermediate results as input and can score the degree of disorder of the permutation. In practice, a scoring model with the architecture of the bert - base model can fit this relationship. Therefore, an attempt is

Table 5: The attack performance (measured by ROUGE-L F1 score %) of heuristic-searching-based cracking after the search curve reaches saturation, using different heuristic signals.

Heuristic Signals	Scoring-model	Frequency (top1-token)	Frequency (top5-token)	(GT) Edit Distance	(GT) Invertion Rouge
Genetic Algorithm	1.02 ± 0.98	8.23 ± 2.44	14.54 ± 3.84	11.45 ± 1.01	28.41 ± 1.98
Gradient-based	6.87 ± 3.21	7.85 ± 2.90	9.87 ± 1.92	-	18.23 ± 2.09

made to use the output score of this model as a heuristic signal.

- Control. As a control, we used two *ground truth* metrics: (a) the mean edit distance (the average edit distance between the cracked permutation matrix and the real permutation matrix), and (b) ROUGE-L of the reconstructed sentence after cracking, compared to the real sentence, as ground truth heuristic signals.

We conducted permutation cracking experiments on an experimental machine equipped with 2 x Intel Xeon Gold 2.60GHz CPUs and 4 x NVIDIA A100(40GB) GPUs. We also recorded the ROUGE-L of the decrypted results of the cracked permutation after the search curve reached saturation (i.e., after the heuristic indicators stopped increasing for a certain period of time). It can be seen from Table 5 that even when using the ground truth (GT) as the heuristic signal for the search, this search task remains difficult (it's hard to break through the 30% performance bottleneck). As mentioned in 2.1, an attacker needs to recover more than 80% of the permutation matrix to achieve an attack performance of over 30%, which is already extremely challenging. Moreover, the heuristic signals adopted by the attacker are perturbed. This perturbation will further confound the features that are already difficult to distinguish as mentioned in Fig. 7, creating an inevitable gap between them and the real signals. This further prevents the recovery performance from surpassing the 30% bottleneck.

D More Efficiency Results

D.1 Communication Overhead Analyses

We analyze the communication overhead of CENTAUR-based PPTI and compare it with the current leading privacy-preserving inference frameworks. For BERT$_{\text{BASE}}$ and BERT$_{\text{LARGE}}$, using CENTAUR for PPTI reduces the communication overhead, respectively, by 2.5 $\sim$ 37.1 and 2.4$\sim$36.0 times compared to existing methods. For the GPT-2$_{\text{BASE}}$ and GPT-2$_{\text{LARGE}}$, this reduction is 2.6$\sim$37.6 and 2.51$\sim$35.4 times, respectively. This significant reduction is attributed to the hybrid computation mechanism employed by CENTAUR, which drastically reduces the communication overhead in both the linear and non-linear layers during PPTI.

Linear Layers. In the linear layers, the communication overhead required for performing PPTI using CENTAUR is half of existing PPTI frameworks. This is because in the baseline PPTI frameworks, both the model parameters and inference data are in secret-sharing states, requiring the use of the private matrix multiplication protocol Π_{MatMul} between secret shares during linear layer operations. In contrast, CENTAUR places only the inference data in a secret-sharing state while keeping the model parameters in a randomly permuted state. This allows CENTAUR to perform most of the linear layer computations using the communication-free private matrix multiplication protocol Π_{ScalMul} between plaintext and secret shares.

Non-Linear Layers. In the non-linear layers, CENTAUR significantly reduces the communication overhead of privacy-preserving computations by converting between secret-sharing and random permutation states. Specifically, for the privacy-preserving computation of Softmax, CENTAUR reduces the communication overhead by 3.1$\sim$112.3 times compared to the current state-of-the-art PPTI frameworks. For the privacy-preserving computation of GeLU, CENTAUR reduces the communication overhead by 2.0$\sim$95.0 times, and for LayerNorm, CENTAUR reduces the communication overhead by 3.0 $\sim$ 3.1 times.

Embedding & Adaptation Layers. The Embedding and Adaptation layers both include linear and nonlinear operations, allowing CENTAUR to achieve dual optimization in communication overhead. Specifically, for the Embedding layer, which includes matrix multiplication and LayerNorm, CENTAUR reduces communication overhead by 22.0 $\sim$27.8 times compared to the current state-of-the-art PPTI frameworks. For the Adaptation layer, CENTAUR reduces communication overhead

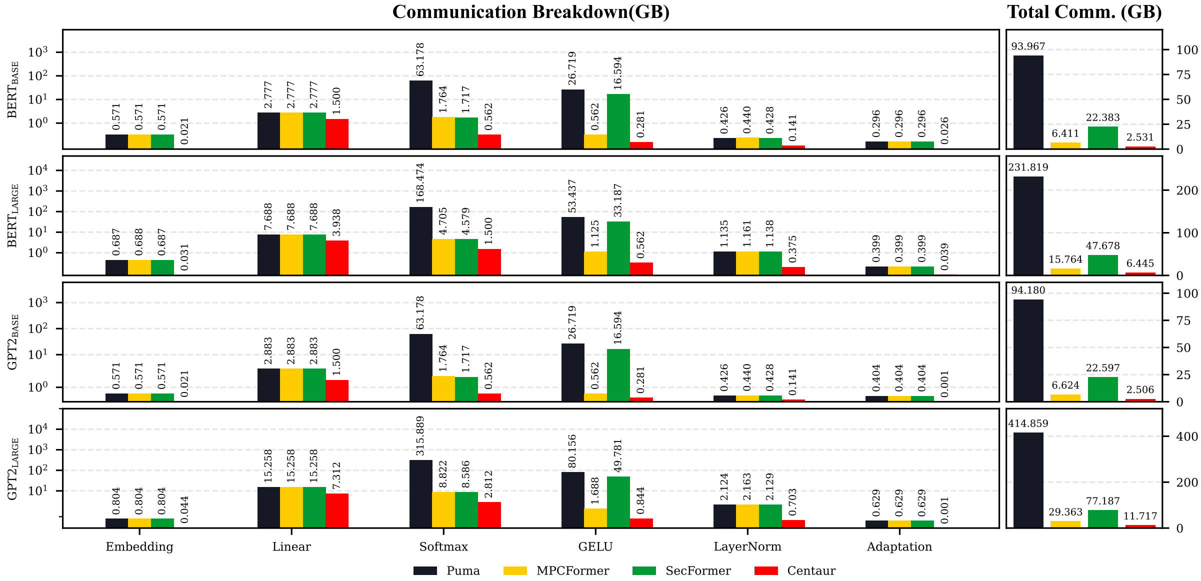


Figure 8: Communication volume for each operations (left) and the entire PPTI process (right) of the tested frameworks.

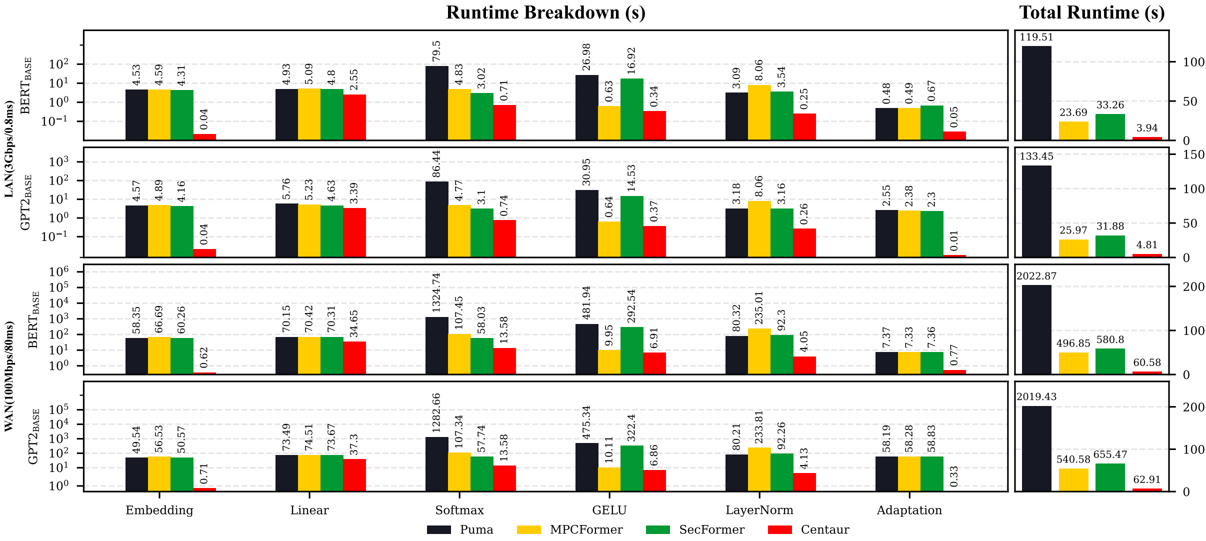


Figure 9: Time breakdown for BERT$_{\text{BASE}}$ and GPT-2$_{\text{BASE}}$. The results are the average of ten runs.

by 10.2 and 11.2 times on the BERT series models. However, for the GPT-2 series models, the reductions are significantly higher, at 448.3 and 698.7 times. This is due to the different structures used in the adaptation layers of BERT and GPT-2 models to adapt to downstream tasks.

D.2 More Time Breakdown Results

In this section, we present the results of the time overhead for privacy-preserving inference using CENTAUR with BERT$_{\text{BASE}}$ and GPT-2$_{\text{BASE}}$ models under LAN and WAN settings. The analysis results are consistent with those observed for BERT$_{\text{LARGE}}$ and GPT-2$_{\text{LARGE}}$ Section 5.4.

E The Generalizability of CENTAUR

CENTAUR is compatible with other Transformer models and can achieve a more optimal balance between privacy, efficiency, and performance. This is due to CENTAUR performing the computation of nonlinear functions in Transformer models under permutation, allowing for seamless extension to other Transformer architectures, such as LLaMA. In particular, the LLaMA model utilizes the RMSNorm normalization function and the SwiGLU activation function. These functions are analogous to LayerNorm and GeLU and can both be computed under permutation, as demonstrated below:

Table 6: The cost of privacy-preserving inference on LLaMA-7B, where #Input denotes the length of the input sentence and #Output represents the number of generated tokens.

(#Input, #Output)	(4, 1)		(8, 1)		(16, 1)	
Costs	Comm(GB)	Time(S)	Comm(GB)	Time(S)	Comm(GB)	Time(S)
CENTAUR	0.32	2.76	0.39	3.02	0.54	6.81

$$\mathrm{RMSNorm}(\mathbf{x}\pi) = \frac{\mathbf{x}\pi}{\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}} = \mathrm{RMSNorm}(\mathbf{x})\pi, \tag{6}$$

$$\mathrm{SwiGLU}(\mathbf{x}\pi) = \frac{\mathbf{x}\pi}{1+e^{-\mathbf{x}\pi}} = \mathrm{SwiGLU}(x)\pi. \tag{7}$$

Where $\mathbf{x} \in \mathbb{R}^d$ is the input vector, d is the input dimension (i.e., the number of elements), and x_i represents the i-th element in the vector. This enables CENTAUR to perform complete privacy-preserving inference on the LLaMA model without changing its underlying architecture. In contrast, for SMPC-based PPTI frameworks, such as MPCFormer and PUMA, extending to the LLaMA model would require the design of proprietary SMPC protocols for handling RMSNorm and SwiGLU. This means that CENTAUR is more generalizable than SMPC-based PPTI frameworks.

Since CENTAUR does not alter the structure of the LLaMA model, it can theoretically achieve performance comparable to the plaintext model. In terms of efficiency, we have further added experimental results of CENTAUR applied to the LLaMA-7B model. We used the same experimental setup as in the paper and executed the experiments in a local area network (LAN) with 20Gbps bandwidth and 0.1ms latency.

From the data in Table 6, it is evident that CENTAUR can complete privacy-preserving inference on the LLaMA-7B model in less than 10 seconds, with communication overheads below 1GB. When the input sequence length is 8, executing privacy-preserving inference on the LLaMA-7B model using CENTAUR generates 1 token in less than 3 seconds, with a communication overhead of 0.39GB. In the same network conditions (bandwidth and latency), PUMA would require approximately 200 seconds and 1.79GB of communication. This shows that CENTAUR offers significant advantages in both speed and communication efficiency, making it a highly scalable and practical solution for privacy-preserving Transformer model inference.

F Hyper-parameter

For the baselines MPCFormer (Li et al., 2023) and SecFormer (Luo et al., 2024), which require additional training and distillation, we followed the fine-tuning and distillation hyperparameter selection method as described in (Li et al., 2023). Specifically, for BERT series models, during the fine-tuning phase, we used learning rates of {1e-6, 5e-6, 1e-5, 1e-4}, batch sizes of {64, 256}, and epochs of {10, 30, 100}. For GPT-2 series models, during the fine-tuning phase, we used learning rates of {1e-6, 5e-6, 1e-5, 1e-4}, a batch size of 2, and epochs of {1, 3, 5}. We fine-tuned each model with these hyperparameter combinations and selected the best-performing model as the teacher.

During the knowledge distillation phase, for BERT series models, the number of distillation iterations was determined based on the MSE loss between the embedding layer and the transformer layer. For small datasets (CoLA, MRPC, RTE), the batch size was 8, while for large datasets (QNLI, STS-B), the batch size was 32. Specifically, for the distillation stages in the embedding layer and transformer layer, QNLI was trained for 10 epochs, MRPC for 20 epochs, STS-B for 50 epochs, CoLA for 50 epochs, and RTE for 50 epochs. For GPT-2 models, we used KLDiv loss to calculate the loss between the output representations of the teacher and student models, and Cosine loss to calculate the loss between the hidden layers of the teacher and student models. The number of distillation steps was determined based on the loss values.