

Private Transformer Inference in MLaaS: A Survey

Yang Li^{1,2}, Xinyu Zhou^{1,2}, Yitong Wang², Liangxin Qian² and Jun Zhao²

¹Energy Research Institute @ NTU, Interdisciplinary Graduate Programme, Nanyang Technological University, Singapore

²College of Computing and Data Science, Nanyang Technological University, Singapore
{yang048, xinyu003, yitong002, qian0080}@e.ntu.edu.sg, junzhao@ntu.edu.sg

Abstract

Transformer models have revolutionized AI, powering applications like content generation and sentiment analysis. However, their deployment in Machine Learning as a Service (MLaaS) raises significant privacy concerns, primarily due to the centralized processing of sensitive user data. Private Transformer Inference (PTI) offers a solution by utilizing cryptographic techniques such as secure multi-party computation and homomorphic encryption, enabling inference while preserving both user data and model privacy. This paper reviews recent PTI advancements (2022–2025), highlighting state-of-the-art solutions and challenges. We also introduce a structured taxonomy and evaluation framework for PTI, focusing on balancing resource efficiency with privacy and bridging the gap between high-performance inference and data privacy.

1 Introduction

Transformer models have emerged as game-changers to revolutionize the field of AI. For instance, both OpenAI ChatGPT and Microsoft Bing have made the power of transformer-based models widely accessible, democratizing advanced AI capabilities. These models leverage attention mechanisms [Vaswani *et al.*, 2017] adeptly to capture long-range dependencies in sequences of input tokens, allowing to model contextual information accurately. Besides, large transformer models are trained on huge quantities of unlabeled textual data and are directly useful for various applications such as sentiment analysis, language translation, content generation, and question answering.

However, applying large transformers still presents privacy risks [Lund and Wang, 2023; Tlili *et al.*, 2023], particularly with the MLaaS model, where servers provide inference services while users supply data (Figure 1). For instance, OpenAI’s ChatGPT operates through an online platform and APIs, requiring users to transmit private data to the server. This reliance raises concerns over data misuse, like unauthorized processing, indefinite storage, or resale to third parties. Even with trustworthy servers, centralized data storage remains vulnerable to breaches and insider threats. Therefore, while MLaaS offers significant convenience and

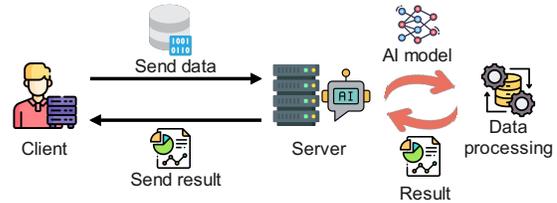


Figure 1: MLaaS without privacy protection

computational power, it also necessitates careful consideration of privacy issues. Such concerns have prompted actions like Italy’s temporary ban of ChatGPT [Mauran, 2023; Lomas, 2023]. The tension between high-performance transformer services and privacy concerns highlights the need for *Private Transformer Inference* (PTI). Private inference is a cryptographic protocol that allows for model inference while ensuring that the server gains no knowledge about the users’ input, and the users learn nothing about the server’s model, apart from inference results. Recently, private inference on transformers has been achieved by using private outsourced computation techniques, such as secure Multi-Party Computation (MPC) [Yao, 1982] and Homomorphic Encryption (HE) [Gentry, 2009]. These advancements make PTI highly promising for enabling privacy-preserving AI in practical fields such as banking and healthcare, facilitating secure data analysis while preserving confidentiality. Platforms like Pyte.ai¹ and Hugging Face’s Private Hub² demonstrate its potential in real-world applications.

To our best knowledge, surveys focusing on PTI do not exist so far, and some recent surveys [Chitty-Venkata *et al.*, 2023; Yan *et al.*, 2024] only investigate optimizing transformer inference or summarize general privacy issues in Large Language Models (LLMs). Notably, we are the first to carefully review the findings of state-of-the-art PTI studies and uniquely discuss the improvements, challenges and present future research directions. The rest of the paper is organized as follows: Section 2 presents preliminaries; Section 3 provides an overview of PTI; Sections 4 and 5 detail solutions to linear and non-linear layers; Section 6 analyzes the experimental results; Section 7 concludes the paper.

¹<https://www.pyte.ai/blog/transformers-based-ai-and-smpc>

²<https://huggingface.co/blog/introducing-private-hub>

2 Preliminaries

2.1 Transformers

Transformer is an encoder-decoder architecture. We focus on the encoder part here, while the decoder can be discussed similarly. An encoder consists of a stack of identical blocks, each with a self-attention layer and a feed-forward network.

Attention layer. Attention layer first maps the input X into three matrices: the query $Q = XW^Q$, the key $K = XW^K$, the value $V = XW^V$, where W^Q, W^K, W^V are learnable weight matrices. The attention score is computed as follows:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V. \quad (1)$$

Multi-head attention can further extend the above mechanism into H different parallel attention layers.

Feed-Forward Layer. A fully connected feed-forward layer consists of two linear transformations with a Gaussian Error Linear Unit (GELU) activation in between:

$$\text{FeedForward}(X) = \text{GELU}(XW_1 + b_1)W_2 + b_2. \quad (2)$$

In addition to the above blocks, an embedding layer is utilized at the beginning of the model, and each layer is wrapped with residual connections and Layer Normalization (LayerNorm).

2.2 Cryptographic Primitives

Various privacy-preserving techniques can be applied to PTI. Here, we focus on two prominent cryptographic primitives: MPC and HE. While MPC and HE may overlap sometimes, we distinguish between them for clarity in this paper.

MPC. MPC enables multiple parties to jointly compute a function over their inputs while keeping those inputs private. It can be formally described as follows: Consider n parties, $(\mathcal{P}_1, \dots, \mathcal{P}_n)$, each holding a private input x_i . The goal is to jointly compute a function $f(x_1, \dots, x_n) \rightarrow y$, where $y = (y_1, \dots, y_n)$. Each party \mathcal{P}_i learns only y (or its portion y_i) without gaining any knowledge of others' inputs. Protocols such as Secret Sharing (SS) and Oblivious Transfer (OT) are commonly used in MPC, and we brief them as follows:

- *Secret Sharing:* SS is a cryptographic technique where a secret s is divided into multiple shares and distributed among n parties. The secret can only be reconstructed when a predefined threshold of shares is combined, and individual shares provide no information about s .
- *Oblivious Transfer:* OT allows a sender \mathcal{S} to securely transmit one of k messages $\{m_0, \dots, m_{k-1}\}$ to a receiver \mathcal{R} , who selects the message using a choice bit $b \in [k]$. The receiver \mathcal{R} learns only m_b , while the sender \mathcal{S} remains unaware of the receiver's choice.

For more details on MPC techniques, readers are suggested to refer to [Evans *et al.*, 2018; Zhao *et al.*, 2019].

HE. HE allows computations on encrypted data without decryption, producing results identical to operations on plaintext after decryption. HE schemes use a public key (pk) for encryption and a secret key (sk) for decryption. The public key can be shared freely, while the secret key remains private. Key homomorphic operations are summarized below.

- $\text{Enc}(\text{pk}, m) \rightarrow c$: On public key pk and a plaintext m , perform encryption to obtain a ciphertext c .
- $\text{Dec}(\text{sk}, c) \rightarrow m$: On secret key sk and a ciphertext c , perform decryption to obtain the plaintext message m .
- $\text{Eval}(\text{pk}, c_0, \dots, c_{k-1}, C) \rightarrow \text{Enc}(\text{pk}, C(m_0, \dots, m_{k-1}))$: Evaluate a circuit C on ciphertexts c_1, \dots, c_k (i.e., encrypted m_1, \dots, m_k), and output the encrypted result of $C(m_0, \dots, m_{k-1})$.

HE schemes could be categorized by the operations used in circuit C and its computational depth. Due to the page limitation, we do not detail them here.

3 An Overview of Private Transformer Inference

In this section, we first review the threat models, summarize the taxonomy of transformer layers in cryptographic contexts, and finally discuss the challenges of PTI.

3.1 Threat Models

The semi-honest (i.e., honest-but-curious) security model is a common assumption in PTI studies. It assumes that all parties adhere to the established protocols honestly but may attempt to extract additional private information passively. This model is typically employed in scenarios where the parties have a foundational level of trust, ensuring they do not actively disrupt the computation. Under this assumption, participants collaboratively contribute to the computation while maintaining protocol integrity. Some studies [Akimoto *et al.*, 2023; Liu and Su, 2024] involve more than two parties in computation. They assume an honest majority setup, where only a small percentage of participants are semi-honest.

Notably, existing PTI studies are less resistant to malicious attacks [Huang and Wang, 2024]. For instance, if a participant maliciously deviates from the protocol, e.g., by refusing to transmit data, the inference process cannot be completed, but data privacy can still be guaranteed.

3.2 Taxonomy of Transformer Layers in Cryptographic Contexts

Based on the required cryptographic operations, transformer layers are categorized into linear and non-linear types. Figure 2 illustrates the architecture of a basic transformer encoder and highlights these categorizations.

- *Linear Layers:* these include embedding, matrix multiplication in attention, and feed-forward layers.
- *Non-linear Layers:* these include Softmax, GELU and LayerNorm.

We will first summarize the efficient solutions to linear layers in Section 4 and then discuss non-linear layers in Section 5.

3.3 Challenges

This subsection examines the challenges of supporting linear and non-linear layers in the context of cryptography.

Linear layers in PTI focus on matrix multiplication, consisting solely of basic additions and multiplications. Hence, they are generally compatible with cryptographic techniques.

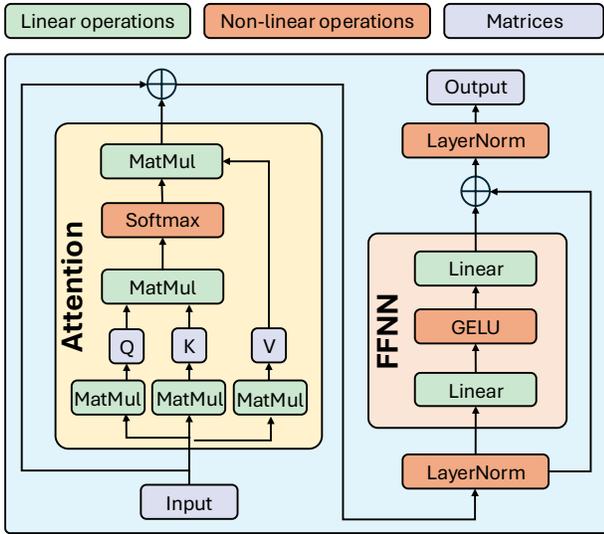


Figure 2: The architecture of a transformer encoder

However, how to efficiently support large matrix multiplications still remains a significant challenge. For example, MPC-based solutions could use techniques such as Beaver triple or OT to support multiplications, but they require substantial data transmission [Lu *et al.*, 2023]. In contrast, HE-based solutions are more communication-efficient, as computations can be performed directly on ciphertexts, but they also bring extensive computational overheads due to the complexity of homomorphic operations.

Non-linear layers often involve complex operations beyond basic arithmetic, e.g., $\exp(x)$ in Softmax. Supporting those operations usually requires specific designs and results in greater overhead than basic addition and multiplication. Hence, non-linear layers often account for the majority of the overall overheads in PTI [Hao *et al.*, 2022; Li *et al.*, 2022; Pang *et al.*, 2024]. For instance, [Li *et al.*, 2022] has provided a baseline using MPC where non-linear layers account for more than 85% of total time (over 50 s). Similarly, [Hao *et al.*, 2022] using HE has also presented that around 60% of its total execution time (over 280 s) comes from non-linear layers. Thereafter, the significant overhead caused by non-linear layers becomes a key bottleneck for PTI.

4 Linear Layers

This section reviews cryptographic protocols for linear layers, focusing on matrix multiplication. The protocols are categorized into two main approaches: MPC-based and HE-based, which will be discussed respectively for clarity.

4.1 MPC-based MatMul Protocols

Most MPC implementations for transformers are based on Additive Secret-Sharing (ASS) schemes. In this case, additions can be evaluated locally for “free”, and multiplications are mainly supported by either Beaver triple [Beaver, 1992] or Replicated Secret Sharing (RSS) [Araki *et al.*, 2016] method.

Beaver triples are random values (a, b, c) such that $a \cdot b = c$, enabling private multiplications by masking the actual inputs during computation. The triples are also split into shares and distributed to computation parties for privacy. Studies [Li *et al.*, 2022; Wang *et al.*, 2022] introduce an additional trusted third-party dealer to generate triples. [Chen *et al.*, 2024] regards the user as a dealer and leaves the computation to multiple servers. Notably, the triples are independent of inputs and thus are usually generated offline to reduce online overheads.

RSS is to split a secret into overlapping shares distributed across parties, enabling private multiplication through local computations and limited exchange of intermediate results. Studies [Dong *et al.*, 2023; Akimoto *et al.*, 2023; Liu and Su, 2024] use the RSS scheme for multiplications. Nonetheless, RSS requires at least three parties throughout the online computation process. This means a typical two-party setup (e.g., a server and client) is insufficient, necessitating the involvement of one third-party participant. In such cases, collusion between any two parties compromises security. Hence, the assumption of an honest majority setup is compulsory, which may affect the practicality of applications.

There is no clearly better scheme in comparison between Beaver triples and RSS, as the choice depends on the setup model and application scenarios. For example, Beaver triples are more efficient during the online phase, whereas RSS performs well on the entire process (offline and online).

4.2 HE-based MatMul Protocols

The use of HE for MatMul in Transformers is relatively unexplored. While MatMul is inherently compatible with HE, as it involves only additions and multiplications, a straightforward element-wise implementation can be highly inefficient [Chen *et al.*, 2022]. There are currently two mainstream ways to accelerate MatMuls in HE:

- 1) Encoding plaintexts into SIMD slots.
- 2) Encoding plaintexts into polynomial coefficients.

The SIMD technique supports batching multiple elements into one ciphertext and enables parallel computation within a single operation, significantly reducing the amortized cost. Studies [Pang *et al.*, 2024; Rovida and Leporati, 2024; Moon *et al.*, 2024; Zhang *et al.*, 2025] have leveraged SIMD in their implementations. However, when applied to MatMul, SIMD requires expensive homomorphic rotations to perform the summation. To address this challenge, [Pang *et al.*, 2024] uses the Baby-Step-Giant-Step (BSGS) to reduce the number of rotations. Similarly, [Zhang *et al.*, 2025] also introduces a slots folding method to reduce the rotation costs.

Studies [Huang *et al.*, 2022; Hao *et al.*, 2022] find that by encoding plaintexts into polynomial coefficients properly, the dot product could directly give the MatMul result and eliminate the need for rotations. Furthermore, [Lu *et al.*, 2023] by Alibaba/Ant proposes a more compact encoding method to alleviate sparsity and save communication. [Hou *et al.*, 2023] customizes a Vector Oblivious Linear Evaluation (VOLE)-based protocol for MatMuls in GPT, reducing the amortized cost of auto-regressively generating response words.

5 Non-Linear Layers

Securing non-linear functions presents a significant challenge due to their cryptographic complexity. This section summarizes approaches in existing PTI studies to efficiently realize non-linear layers, including Softmax, GELU and LayerNorm.

5.1 Softmax

Softmax is to perform a re-weight normalization of the obtained attention map. Given an input vector $\mathbf{x} = [x_i |_{i \in [d]}]$, the Softmax function on each x_i can be formulated as:

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_{j=1}^d \exp(x_j)}, \quad (3)$$

where the main challenge is efficiently calculating the underlying exponential function $\exp(x)$ and the reciprocal computation $1/x$. The reciprocal computation $1/x$ is relatively well-established and is often treated as a black-box operation [Dong *et al.*, 2023]. Consequently, recent PTI studies have focused on optimizing the computation of $\exp(x)$.

Several studies employ aggressive crypto-friendly functions to directly replace the $\exp(x)$ in Softmax, which often brings considerable efficiency at the cost of accuracy. Specifically, we present some of their methods as follows:

$$\exp(x) \sim \begin{cases} (x+c)^2, & [\text{Li } et al., 2022; \text{Luo } et al., 2024] \\ \text{RELU}(x), & [\text{Zeng } et al., 2023] \\ (ax+c)^2, & [\text{Zhang } et al., 2023] \\ (x+c)^4. & [\text{Chen } et al., 2023] \end{cases} \quad (4)$$

It is obvious that substitutions in (4) would make the approximated Softmax differ a lot by numerical values. Hence, the above studies utilize the Knowledge Distillation (KD) [Hinton *et al.*, 2006] method to bridge the performance gap. However, KD depends on a well-trained teacher model and adds extra computational overhead, which may not be practical.

To maintain the accuracy and eliminate the need for KD, another mainstream solution is to design polynomial approximations for $\exp(x)$. Notably, a key feature of the $\exp(x)$ function is its susceptibility to instability when handling large input values. Hence, most studies will adjust the input range before applying approximations to ensure both stability and computational efficiency. For instance, [Rovida and Leporati, 2024] first scales the input into a small range of $[-1, 1]$ and then approximates $\exp(x)$ using the Maclaurin series:

$$\exp(x) \approx \sum_{i=0}^6 \frac{x^i}{i!}. \quad (5)$$

To further improve the accuracy and numerical stability, studies [Dong *et al.*, 2023; Lu *et al.*, 2023; Hou *et al.*, 2023; Zhang *et al.*, 2025] replace the original input x_i with $x_i - \max(\mathbf{x})$ to ensure non-positivity, and then design piecewise polynomials with Taylor Series for approximation:

$$\exp(x) \approx \begin{cases} 0, & \text{if } x \leq a \\ (1 + \frac{x}{2^r})^{2^r}, & \text{if } a \leq x \leq 0 \end{cases} \quad (6)$$

where the values of threshold a and polynomial order r differ in the above studies. Notably, one should be careful about the choice of r ; too high a polynomial order can bring better accuracy but decrease computational efficiency.

5.2 GELU

GELU in Transformers provides smooth and non-linear activation for modeling complex patterns. The GELU activation for an input x is defined as follows:

$$\text{GeLU}(x) = \frac{x}{2} \left(1 + \text{erf}\left(\frac{x}{\sqrt{2}}\right)\right), \quad (7)$$

where $\text{erf}(\cdot)$ is the Gaussian error function, expressed as $\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$. Similar to $\text{Softmax}(x)$, solutions to $\text{GeLU}(x)$ in cryptographic context can be mainly categorized into three methods: substitution, approximation, and LUT.

Studies [Chen *et al.*, 2022; Park *et al.*, 2024] directly replace GELU with crypto-friendly RELU (i.e., $\max(0, x)$) since support for comparison operations in cryptography is relatively well established [Cheon *et al.*, 2020; Huang *et al.*, 2022]. MPCFormer [Li *et al.*, 2022] proposes a more aggressive substitution with a quadratic function:

$$\text{GELU}(x) \sim 0.125x^2 + 0.25x + 0.5. \quad (8)$$

Other studies [Dong *et al.*, 2023; Lu *et al.*, 2023; Pang *et al.*, 2024; Zhang *et al.*, 2025] rely on the polynomials to approximate GELU. Since the GELU function is almost linear with a larger or smaller input, they suggest an efficient low-degree polynomial (e.g., $n \leq 4$ in [Pang *et al.*, 2024]) for approximation within the short interval around 0. A general expression for approximation is shown below:

$$\text{GELU}(x) \approx \begin{cases} -c, & \text{if } x < a \\ \sum_{i=0}^n a_i x^i, & \text{if } a \leq x \leq b \\ x - c, & \text{if } b < x \end{cases} \quad (9)$$

where $[a, b]$ is a small interval around 0, c is a small non-negative number (often 0), and a_i denote the obtained polynomial coefficient from different approximation methods.

5.3 LayerNorm

LayerNorm ensures that inputs across different layers have a consistent mean and variance to enhance stability. For a given vector $\mathbf{x} \in \mathbb{R}^d$, the LayerNorm function is defined as follows:

$$\text{LayerNorm}(x_i) = \frac{(x_i - \mu)}{\sigma} \cdot \gamma + \beta, \quad (10)$$

where $\mu = \sum_{i=1}^d x_i/d$ and $\sigma = \sqrt{\sum_{i=1}^d (x_i - \mu)^2}$ are mean and standard deviation, and γ and β are affine transform parameters. The main challenge lies in the required reciprocal square root operation of σ .

One intuitive solution is to employ well-established protocols for $1/\sqrt{x}$. Studies [Ding *et al.*, 2023; Zhang *et al.*, 2025; Park *et al.*, 2024] employ Newton-like methods [Qu and Xu, 2023] to iteratively compute $1/\sqrt{x}$. Similarly, [Luo *et al.*, 2024] uses Goldschmidt's method [Markstein, 2004] to convert square root inverses into iterations of multiplications. Due to page limitations, we don't detail the protocols here. In general, the above solutions could maintain the accuracy of LayerNorm but at the cost of considerable overhead.

Some studies manage to avoid non-linear computations by altering the architecture of LayerNorm. For instance, [Chen

et al., 2022] directly removes μ and σ , leaving the mean and standard deviation achieved by learnable parameters γ and β :

$$\text{LayerNorm}(x_i) \sim x_i \cdot \gamma + \beta. \quad (11)$$

Similarly, [Liu and Liu, 2023] removes the standard deviation part and then re-trains the model. In a different way, [Rovida and Leporati, 2024] experimentally observes the values of μ and σ to simplify the computation as follows:

$$\text{LayerNorm}(x_i) \approx (x_i - E_p) \cdot (V_p \gamma) + \beta, \quad (12)$$

where E_p and V_p are precomputed values for μ and $1/\sigma$. Notably, these methods simplify computations at the expense of LayerNorm accuracy, and therefore often require re-training to recover model accuracy.

6 Reported Experiments

This section compares the reported experimental results presented in selected studies. A summary of their evaluations is presented in Table 1.

6.1 Models and Datasets

Popular transformers for PTI studies mainly include the BERT family (e.g., Bert-Tiny, Bert-Base, Roberta-Base) and the GPT family (e.g., GPT2-Base). Some studies have also attempted implementing other transformer models, such as LLaMA-7B [Touvron *et al.*, 2023], ViT-Base [Dosovitskiy *et al.*, 2020], and its variant CCT [Hassani *et al.*, 2021]. Transformers with more complex structures and a larger number of parameters tend to take longer to do inference, but they often perform better [Jiao *et al.*, 2019].

Most PTI studies evaluate performance using the GLUE benchmark [Wang *et al.*, 2018], a standard for BERT and GPT-based transformers, which requires models to process single-sentence and sentence-pair inputs for predictions. It contains three NLU tasks and nine corresponding corpora: single-sentence tasks (CoLA, SST-2), similarity and paraphrase tasks (MRPC, STS-B, QQP), and inference tasks (MNLI, QNLI, RTE, WNLI). Studies [Dong *et al.*, 2023; Zhang *et al.*, 2025] evaluate GPT2 on Wikitext-103 V1 [Merity *et al.*, 2016] and CBT-CN. ViT-Base is designed for image processing, and thus BumbleBee [Lu *et al.*, 2023] uses the ImageNet dataset for its evaluation. A series of studies [Zeng *et al.*, 2023; Zhang *et al.*, 2023; Chen *et al.*, 2023] focusing on CCT are evaluated on CIFAR and Tiny-ImageNet.

6.2 Communication Overhead

According to Table 1, it is obvious that the studies employing MPC usually require considerable communication overhead. This is because their cornerstone MPC techniques (e.g., SS and OT) all require multiple rounds of communication between parties for multiplication and other nonlinear operations, thus causing significant communication costs. Early study [Hao *et al.*, 2022] using ASS requires 280.99 GB for evaluating BERT-Base, while study [Gupta *et al.*, 2023] requires only 0.99 GB on the same model by using a more efficient Function Secret Sharing (FSS) scheme.

In contrast, studies only using HE techniques [Rovida and Leporati, 2024; Zimmerman *et al.*, 2024; Zhang *et al.*, 2025] often require minimum communication. Once the encrypted input from the client is transmitted to the server, the remaining computation will be completed by the server only, which is a non-interactive process. For instance, NEXUS [Zhang *et al.*, 2025] requires only 0.16 GB for evaluating the BERT-Base model, which is lower than any other MPC studies. Other studies only using HE do not report the communication overhead, as the main bottleneck in them comes from computation. We will detail this in the next subsection.

6.3 Runtime

To ensure fairness, we report the runtime performance of selected studies on CPU only in Table 1, even though some of them also support GPU implementation. However, as most of these studies are not open source, differences in experimental platforms may still affect the runtime performance. Due to page limitations, we only provide their communication setups for readers’ reference, and more detailed computation setups can be found in their manuscripts.

Overall, PTI solutions relying only on MPC tend to achieve faster runtimes than HE-only and hybrid (MPC+HE) ones. Those solutions mainly use secret-sharing schemes with low computation complexity but high communication overheads. Consequently, they often deliver the fastest runtimes, particularly in communication-rich environments such as LAN settings. Under a setup with a bandwidth of 9.4 Gbps and a latency of 0.05 ms, [Gupta *et al.*, 2023] demonstrated that BERT-Base could be evaluated in just 1.84 s. Hybrid solutions use HE for linear computations to reduce communication overhead, but they also increase runtime due to the computational complexity of homomorphic operations. For instance, [Pang *et al.*, 2024] evaluates BERT-Base in 185 s. HE-only solutions require the longest runtime. Latest studies [Zimmerman *et al.*, 2024; Moon *et al.*, 2024; Zhang *et al.*, 2025] all require over 400 s for evaluation on the BERT-Base model. Even for the smaller BERT-Tiny model, [Rovida and Leporati, 2024] still requires over 200 s. In particular, the most consuming part of those studies is the *Bootstrapping* operation, which is often used to “refresh” a ciphertext to reduce noise. The bootstrapping part in [Zhang *et al.*, 2025] and [Moon *et al.*, 2024] accounts for 37.72% and 53.96% of the total runtime, respectively.

6.4 Accuracy

This subsection reviews the accuracy performance of different PTI solutions. Specifically, linear computations (e.g., matrix multiplication) are often well supported by cryptographic techniques. Hence, most of the accuracy drop reported in Table 1 comes from the treatment of non-linear layers. For example, an early study [Li *et al.*, 2022] employed aggressive substitutions for the Softmax and GELU functions, leading to a substantial accuracy drop of 8.8% on the STS-B dataset. In contrast, a more recent study [Pang *et al.*, 2024] utilized refined polynomial approximations, reducing the accuracy loss to just 1.18% on the same dataset. Notably, knowledge distillation techniques could effectively cure the accuracy drop due to approximation operations. Studies [Zeng *et al.*, 2023;

Table 1: Resource requirements, tasks performed, dataset performance, and runtime.

Study	Techniques	Model	Dataset	Comm.	Comm. Set.	Performance			Runtime
						Plain	Enc.	Loss ↓	
[Li <i>et al.</i> , 2022]	MPC	BERT-Base	QNLI MRPC RTE STS-B	12.089 GB	(5 Gbps, 1 ms)	91.7 % 90.3 % 69.7 % 89.1 %	90.6 % 88.7 % 64.9 % 80.3 %	1.1 % 1.6 % 4.8 % 8.8 %	55.320 s
[Dong <i>et al.</i> , 2023]	MPC	BERT-Base	CoLA RTE QNLI	10.773 GB	(5 Gbps, 1 ms)	61.6 % 70.0 % 91.6 %	61.3 % 70.0 % 91.6 %	0.3 % 0.0 % 0.0 %	33.913 s
		Roberta-Base	-	11.463 GB		-	-	-	41.641 s
		Bert-Large	-	27.246 GB		-	-	-	73.720 s
		GPT2-Base	Wiki.-103	3.774 GB		16.284	16.284	0.000	15.506 s
		GPT2-Medium		7.059 GB		12.536	12.540	-0.004	30.272 s
		GPT2-Large		11.952 GB		10.142	10.161	-0.019	54.154 s
LLaMA-7B	-	1.794 GB		-	-	-	200.473 s		
[Gupta <i>et al.</i> , 2023]	MPC	BERT-Tiny	SST2 MRPC QNLI	0.02 GB	(9.4 Gbps, 0.05 ms)	81.19 % 72.54 % 81.64 %	81.42 % 72.79 % 81.73 %	-0.23 % -0.25 % -0.09 %	0.09 s
		BERT-Base	SST2 MRPC QNLI	0.99 GB		90.59 % 84.31 % 88.72 %	90.25 % 83.82 % 89.03 %	0.34 % 0.49 % -0.31 %	1.84 s
		BERT-Large	SST2 MRPC QNLI	2.63 GB		88.99 % 78.67 % 92.23 %	88.99 % 78.92 % 92.31 %	0.0 % -0.25 % -0.08 %	4.73 s
		GPT-2	Lambada	0.82 GB		32.46 %	33.28 %	-0.82 %	1.61 s
		GPT-Neo	Lambada	4.02 GB		57.46 %	57.81 %	-0.35 %	7.43 s
		LLaMA2-7B	Lambada	12.35 GB		70.17 %	70.01 %	0.16 %	27.01 s
		LLaMA2-13B	Lambada	19.33 GB		73.14 %	72.98 %	0.16 %	44.13 s
		[Zheng <i>et al.</i> , 2023]	MPC	BERT-Tiny		MRPC SST-2	0.9 GB	(100 Mbps, 2.3 ms)	-
BERT-Base	SST2 QNLI	3.6 GB	-	86.3 % 92.5 %	-	35.4 s			
BERT-Large	SST2 QNLI	7.9 GB	-	87.6 % 93.5 %	-	91.6 s			
[Zeng <i>et al.</i> , 2023]	MPC	CCT	CIFAR-10 CIFAR-100 Tiny-ImageNet	-	(44 Mbps, 40 ms)	95.56 % 77.36 % 61.60 %	94.27 % 77.76 % 63.45 %	1.29 % -0.40 % -2.35 %	50.94 s 51.33 s 74.34 s
[Zhang <i>et al.</i> , 2023]	MPC	CCT	CIFAR-10 CIFAR-100 Tiny-ImageNet	-	-	95.56 % 77.36 % 61.60 %	95.92 % 77.86 % 63.49 %	-0.36 % -0.50 % -2.39 %	40.65 s 38.97 s 66.21 s
[Chen <i>et al.</i> , 2023]	MPC	CCT	CIFAR-10 CIFAR-100 Tiny-ImageNet	-	-	95.56 % 77.36 % 61.60 %	94.97 % 79.04 % 65.86 %	0.59 % -1.68 % -4.26 %	14.04 s 14.13 s 44.27 s
[Luo <i>et al.</i> , 2024]	MPC	BERT-Base	MRPC STS-B	23.593 GB	(10 Gbps)	90.3 % 89.1 %	89.2 % 87.4 %	1.1 % 1.7 %	19.513 s
		BERT-Large	MRPC STS-B	50.364 GB		90.6 % 90.2 %	88.7 % 89.2 %	1.9 % 1.0 %	39.089 s
[Hao <i>et al.</i> , 2022]	MPC+HE	BERT-Base	SST-2 MRPC STS-B	280.99 GB	(3 Gbps, 0.8 ms)	92.36 % 90.00 % 89.62 %	92.77 % 89.87 % 89.41 %	-0.41 % 0.13 % 0.21 %	475 s
[Lu <i>et al.</i> , 2023]	MPC+HE	BERT-Base	QNLI RTE CoLA	-	(1 Gbps, 0.5 ms)	90.30 % 70.04 % 61.57 %	90.20 % 70.04 % 60.82 %	0.10 % 0.00 % 0.75 %	-
		BERT-Large	-	20.85 GB		-	-	-	404.4 s
		LLaMA-7B	-	6.82 GB		-	-	-	832.2 s
		GPT2-Base	-	1.94 GB		-	-	-	55.2 s
		ViT-Base	ImageNet	14.44 GB		89.44 %	89.13 %	0.31 %	234 s
[Pang <i>et al.</i> , 2024]	MPC+HE	BERT-Base	SST-2 MRPC RTE STS-B	25.74 GB	(3 Gbps, 0.8 ms)	92.36 % 90.00 % 69.70 % 89.62 %	92.78 % 89.95 % 69.31 % 88.44 %	-0.42 % 0.05 % 0.39 % 1.18 %	185 s
[Chen <i>et al.</i> , 2022]	HE	BERT-Tiny	SST-2 STS-B	-	-	82.45 % 72.83 %	82.11 % 68.39 %	0.34 % 4.44 %	≈ 4700 s
[Rovida and Leporati, 2024]	HE	BERT-Tiny	SST-2	-	-	83.7 %	79.0 %	4.7 %	214 s
[Moon <i>et al.</i> , 2024]	HE	BERT-Base	MRPC	-	-	85.29 %	84.80 %	0.49 %	625.8 s
[Zimmerman <i>et al.</i> , 2024]	HE	Roberta-Base	SST-2 QNLI MNLI	-	-	94.80 % 92.80 % 87.60 %	93.35 % 91.62 % 86.93 %	1.45 % 1.18 % 0.67 %	≈ 400 s
[Zhang <i>et al.</i> , 2025]	HE	BERT-Base	RTE SST-2 QNLI	0.16 GB	(100 Mbps, 80 ms)	70.04 % 92.36 % 90.30 %	69.88 % 92.11 % 89.92 %	0.16 % 0.25 % 0.38 %	857 s
		LLaMA-3B	RTE SST-2 QNLI			82.75 % 94.94 % 90.70 %	81.24 % 94.46 % 90.20 %	1.51 % 0.48 % 0.50 %	1088 s

Zhang *et al.*, 2023; Chen *et al.*, 2023] using knowledge distillation even achieved better performances than the plaintext model on CIFAR-100 and Tiny-ImageNet datasets.

7 Conclusion and Future Directions

In this paper, we have comprehensively reviewed the existing cryptographic solutions for private transformer inference. To the best of our knowledge, this is the first review to categorize these solutions within a structured taxonomy, providing an in-depth discussion. In addition, we also offer a detailed performance comparison of various methods from different perspectives. Despite these advancements, several open challenges remain unresolved, which we summarize as follows:

GPU Acceleration. One important limitation of current PTI solutions is the lack of GPU acceleration support. Cryptographic operations, particularly those in HE, are computationally expensive. Hence, leveraging GPUs for those operations could dramatically improve the private inference speed [Zhang *et al.*, 2025]. Most PTI studies rely on popular CPU-based cryptographic libraries (e.g., EzPC and SEAL) for implementations. Only a few studies [Gupta *et al.*, 2023; Zhang *et al.*, 2025] have explored GPU-accelerated solutions, achieving significant speedups of $15.0\times$ and $22.9\times$ on the BERT-Base model compared to CPU implementations. In recent years, a number of cryptographic libraries supporting GPU accelerations have been released, e.g., [Ma *et al.*, 2023; Wang *et al.*, 2023; Jawalkar *et al.*, 2024; Yang *et al.*, 2024], showing great potential for improving runtime. We believe future works should explore optimizing GPU-based frameworks for PTI to make real-time inference more feasible.

Scalability on Generation Tasks. Another concern is that existing PTI solutions are seldom evaluated on content generation tasks, e.g., translation and Q&A. As shown in Table 1, most studies have focused on classification tasks (e.g., GLUE and ImageNet) and achieved promising results. However, cryptographic solutions for transformers introduce approximation errors that vary by task. In classification tasks, evaluation primarily depends on identifying the highest category scores, where the errors may have minimal effect on overall accuracy [Rovida and Leporati, 2024]. Generation tasks could be more sensitive to such errors, as they can lead to nonsensical outputs and severely degrade performance. This issue was also highlighted in the open review³ of PUMA [Dong *et al.*, 2023]. An example of a failed Q&A is as follows:

- Q: What is the largest animal?
- A: * a. What is the difference between? Sedition between? Sedition ? ? between

Thereafter, expanding PTI evaluations to include diverse generative applications (e.g., dialogue systems and creative writing tasks) will be essential for realizing their potential.

Practicality in Real-World Applications. Despite their theoretical promise, the practicality of PTI solutions in real-world applications remains a significant challenge. Current implementations often involve high computational and memory overheads, making deployment on edge devices or in latency-sensitive scenarios impractical. Additionally, many

cryptographic methods require extensive parameter tuning and complex setup, further hindering their usability. Future research should prioritize developing lightweight and user-friendly PTI frameworks that balance security and performance. Moreover, integrating these solutions into widely used AI platforms could facilitate their adoption in industries such as healthcare, finance, and personalized AI services, where privacy is paramount. Addressing these concerns will be crucial for bridging the gap between theoretical advancements and practical deployments.

Ethical Statement

Our experiments use only open-source datasets and do not involve human participants or sensitive data. While our work promotes secure AI deployment, we acknowledge potential misuse and advocate for responsible AI practices.

References

- [Akimoto *et al.*, 2023] Yoshimasa Akimoto, Kazuto Fukuchi, Youhei Akimoto, and Jun Sakuma. Privacy-preserving transformer with MPC. In *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*, pages 392–410. IEEE, 2023.
- [Araki *et al.*, 2016] Toshinori Araki, Jun Furukawa, Yehuda Lindell, Ariel Nof, and Kazuma Ohara. High-throughput semi-honest secure three-party computation with an honest majority. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 805–817, 2016.
- [Beaver, 1992] Donald Beaver. Efficient multiparty protocols using circuit randomization. In *Advances in Cryptology—CRYPTO’91: Proceedings 11*, pages 420–432. Springer, 1992.
- [Chen *et al.*, 2022] Tianyu Chen, Hangbo Bao, Shaohan Huang, Li Dong, Binxing Jiao, Daxin Jiang, Haoyi Zhou, Jianxin Li, and Furu Wei. The-x: Privacy-preserving transformer inference with homomorphic encryption. *arXiv preprint arXiv:2206.00216*, 2022.
- [Chen *et al.*, 2023] Dake Chen, Yuke Zhang, Souvik Kundu, Chenghao Li, and Peter A Beerel. RNA-ViT: Reduced-dimension approximate normalized attention vision transformers for latency efficient private inference. In *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, pages 1–9. IEEE, 2023.
- [Chen *et al.*, 2024] Yuntian Chen, Xianjia Meng, Zhiying Shi, Zhiyuan Ning, and Jingzhi Lin. SecureTLM: Private inference for transformer-based large model with MPC. *Information Sciences*, 667:120429, 2024.
- [Cheon *et al.*, 2020] Jung Hee Cheon, Dongwoo Kim, and Duhyeong Kim. Efficient homomorphic comparison methods with optimal complexity. In *Advances in Cryptology—ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part II 26*, pages 221–256. Springer, 2020.

³<https://openreview.net/forum?id=x3LxHdZX0f>

- [Chitty-Venkata *et al.*, 2023] Krishna Teja Chitty-Venkata, Sparsh Mittal, Murali Emani, Venkatram Vishwanath, and Arun K. Somani. A survey of techniques for optimizing transformer inference. *Journal of Systems Architecture*, 144:102990, 2023.
- [Ding *et al.*, 2023] Yuanchao Ding, Hua Guo, Yewei Guan, Weixin Liu, Jiarong Huo, Zhenyu Guan, and Xiyong Zhang. East: Efficient and accurate secure transformer framework for inference. *arXiv preprint arXiv:2308.09923*, 2023.
- [Dong *et al.*, 2023] Ye Dong, Wen-jie Lu, Yancheng Zheng, Haoqi Wu, Derun Zhao, Jin Tan, Zhicong Huang, Cheng Hong, Tao Wei, and Wenguang Cheng. Puma: Secure inference of Llama-7b in five minutes. *arXiv preprint arXiv:2307.12533*, 2023.
- [Dosovitskiy *et al.*, 2020] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- [Evans *et al.*, 2018] David Evans, Vladimir Kolesnikov, Mike Rosulek, et al. A pragmatic introduction to secure multi-party computation. *Foundations and Trends® in Privacy and Security*, 2(2-3):70–246, 2018.
- [Gentry, 2009] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 169–178, 2009.
- [Gupta *et al.*, 2023] Kanav Gupta, Neha Jawalkar, Ananta Mukherjee, Nishanth Chandran, Divya Gupta, Ashish Panwar, and Rahul Sharma. SIGMA: secure GPT inference with function secret sharing. *Cryptology ePrint Archive*, 2023.
- [Hao *et al.*, 2022] Meng Hao, Hongwei Li, Hanxiao Chen, Pengzhi Xing, Guowen Xu, and Tianwei Zhang. Iron: Private inference on transformers. *Advances in neural information processing systems*, 35:15718–15731, 2022.
- [Hassani *et al.*, 2021] Ali Hassani, Steven Walton, Nikhil Shah, Abulikemu Abuduweili, Jiachen Li, and Humphrey Shi. Escaping the big data paradigm with compact transformers. *arXiv preprint arXiv:2104.05704*, 2021.
- [Hinton *et al.*, 2006] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.
- [Hou *et al.*, 2023] Xiaoyang Hou, Jian Liu, Jingyu Li, Yuhan Li, Wen-jie Lu, Cheng Hong, and Kui Ren. Ciphert: Secure two-party GPT inference. *Cryptology ePrint Archive*, 2023.
- [Huang and Wang, 2024] Hai Huang and Yongjian Wang. SecBERT: Privacy-preserving pre-training based neural network inference system. *Neural Networks*, 172:106135, 2024.
- [Huang *et al.*, 2022] Zhicong Huang, Wen-jie Lu, Cheng Hong, and Jiansheng Ding. Cheetah: Lean and fast secure {Two-Party} deep neural network inference. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 809–826, 2022.
- [Jawalkar *et al.*, 2024] Neha Jawalkar, Kanav Gupta, Arkaprava Basu, Nishanth Chandran, Divya Gupta, and Rahul Sharma. Orca: Fss-based secure training and inference with GPUs. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 597–616. IEEE, 2024.
- [Jiao *et al.*, 2019] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.
- [Li *et al.*, 2022] Dacheng Li, Hongyi Wang, Rulin Shao, Han Guo, Eric Xing, and Hao Zhang. MPCFORMER: Fast, performant and private transformer inference with MPC. In *The Eleventh International Conference on Learning Representations*, 2022.
- [Liu and Liu, 2023] Xuanqi Liu and Zhuotao Liu. LLMs can understand encrypted prompt: Towards privacy-computing friendly transformers. *arXiv preprint arXiv:2305.18396*, 2023.
- [Liu and Su, 2024] Yanxin Liu and Qianqian Su. PPTIF: Privacy-preserving transformer inference framework for language translation. *IEEE Access*, 2024.
- [Lomas, 2023] Natasha Lomas. Italy orders ChatGPT blocked citing data protection concerns. *TechCrunch, March*, 31, 2023.
- [Lu *et al.*, 2023] Wen-jie Lu, Zhicong Huang, Zhen Gu, Jingyu Li, Jian Liu, Kui Ren, Cheng Hong, Tao Wei, and Wenguang Chen. Bumblebee: Secure two-party inference framework for large transformers. *Cryptology ePrint Archive*, 2023.
- [Lund and Wang, 2023] Brady D Lund and Ting Wang. Chatting about ChatGPT: how may AI and GPT impact academia and libraries? *Library hi tech news*, 40(3):26–29, 2023.
- [Luo *et al.*, 2024] Jinglong Luo, Yehong Zhang, Zhuo Zhang, Jiaqi Zhang, Xin Mu, Hui Wang, Yue Yu, and Zenglin Xu. SecFormer: Fast and accurate privacy-preserving inference for transformer models via SMPC. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 13333–13348, 2024.
- [Ma *et al.*, 2023] Junming Ma, Yancheng Zheng, Jun Feng, Derun Zhao, Haoqi Wu, Wenjing Fang, Jin Tan, Chaofan Yu, Benyu Zhang, and Lei Wang. {SecretFlow-SPU}: A performant and {User-Friendly} framework for {Privacy-Preserving} machine learning. In *2023 USENIX Annual Technical Conference (USENIX ATC 23)*, pages 17–33, 2023.
- [Markstein, 2004] Peter Markstein. Software division and square root using Goldschmidt’s algorithms. In *Proceedings of the 6th Conference on Real Numbers and Computers (RNC’6)*, volume 123, pages 146–157. Citeseer, 2004.
- [Mauran, 2023] Cecily Mauran. Whoops, samsung workers accidentally leaked trade secrets via ChatGPT.

- Mashable* [online]. Dostupné z: <https://mashable.com/article/samsungchatgpt-leak-details>, 2023.
- [Merity *et al.*, 2016] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.
- [Moon *et al.*, 2024] Jungho Moon, Dongwoo Yoo, Xiaoqian Jiang, and Miran Kim. THOR: Secure transformer inference with homomorphic encryption. *Cryptology ePrint Archive*, 2024.
- [Pang *et al.*, 2024] Qi Pang, Jinhao Zhu, Helen Möllering, Wenting Zheng, and Thomas Schneider. BOLT: Privacy-preserving, accurate and efficient inference for transformers. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 130–130. IEEE Computer Society, 2024.
- [Park *et al.*, 2024] Dongjin Park, Eunsang Lee, and Joon-woo Lee. Powerformer: Efficient privacy-preserving transformer with batch rectifier-power max function and optimized homomorphic attention. *Cryptology ePrint Archive*, 2024.
- [Qu and Xu, 2023] Hongyuan Qu and Guangwu Xu. Improvements of homomorphic secure evaluation of inverse square root. In *International Conference on Information and Communications Security*, pages 110–127. Springer, 2023.
- [Rovida and Loporati, 2024] Lorenzo Rovida and Alberto Loporati. Transformer-based language models and homomorphic encryption: An intersection with bert-tiny. In *Proceedings of the 10th ACM International Workshop on Security and Privacy Analytics*, pages 3–13, 2024.
- [Tlili *et al.*, 2023] Ahmed Tlili, Boulus Shehata, Michael Agyemang Adarkwah, Aras Bozkurt, Daniel T Hickey, Ronghuai Huang, and Brighter Agyemang. What if the devil is my guardian angel: ChatGPT as a case study of using chatbots in education. *Smart learning environments*, 10(1):15, 2023.
- [Touvron *et al.*, 2023] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- [Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [Wang *et al.*, 2018] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.
- [Wang *et al.*, 2022] Yongqin Wang, G Edward Suh, Wenjie Xiong, Benjamin Lefaudeaux, Brian Knott, Murali Annavaram, and Hsien-Hsin S Lee. Characterization of MPC-based private inference for transformer-based models. In *2022 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 187–197. IEEE, 2022.
- [Wang *et al.*, 2023] Zhiwei Wang, Peinan Li, Rui Hou, Zhihao Li, Jiangfeng Cao, XiaoFeng Wang, and Dan Meng. HE-Booster: an efficient polynomial arithmetic acceleration on gpus for fully homomorphic encryption. *IEEE Transactions on Parallel and Distributed Systems*, 34(4):1067–1081, 2023.
- [Yan *et al.*, 2024] Biwei Yan, Kun Li, Minghui Xu, Yueyan Dong, Yue Zhang, Zhaochun Ren, and Xiuzhen Cheng. On protecting the data privacy of large language models (LLMs): A survey. *arXiv preprint arXiv:2403.05156*, 2024.
- [Yang *et al.*, 2024] Hao Yang, Shiyu Shen, Wangchen Dai, Lu Zhou, Zhe Liu, and Yunlei Zhao. Phantom: a cuda-accelerated word-wise homomorphic encryption library. *IEEE Transactions on Dependable and Secure Computing*, 2024.
- [Yao, 1982] Andrew C Yao. Protocols for secure computations. In *23rd annual symposium on foundations of computer science (sfcs 1982)*, pages 160–164. IEEE, 1982.
- [Zeng *et al.*, 2023] Wenxuan Zeng, Meng Li, Wenjie Xiong, Tong Tong, Wen-jie Lu, Jin Tan, Runsheng Wang, and Ru Huang. Mpcvit: Searching for accurate and efficient MPC-friendly vision transformer with heterogeneous attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5052–5063, 2023.
- [Zhang *et al.*, 2023] Yuke Zhang, Dake Chen, Souvik Kundu, Chenghao Li, and Peter A Beerel. 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, 2023.
- [Zhang *et al.*, 2025] Jiawen Zhang, Jian Liu, Xinpeng Yang, Yinghao Wang, Kejia Chen, Xiaoyang Hou, Kui Ren, and Xiaohu Yang. Secure transformer inference made non-interactive. *Network and Distributed System Security (NDSS) Symposium*, 2025.
- [Zhao *et al.*, 2019] Chuan Zhao, Shengnan Zhao, Minghao Zhao, Zhenxiang Chen, Chong-Zhi Gao, Hongwei Li, and Yu-an Tan. Secure multi-party computation: theory, practice and applications. *Information Sciences*, 476:357–372, 2019.
- [Zheng *et al.*, 2023] Mengxin Zheng, Qian Lou, and Lei Jiang. Primer: Fast private transformer inference on encrypted data. In *2023 60th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2023.
- [Zimerman *et al.*, 2024] Itamar Zimerman, Allon Adir, Ehud Aharoni, Matan Avitan, Moran Baruch, Nir Drucker, Jenny Lerner, Ramy Masalha, Reut Meiri, and Omri Soceanu. Power-softmax: Towards secure LLM inference over encrypted data. *arXiv preprint arXiv:2410.09457*, 2024.