Primer: Fast Private Transformer Inference on Encrypted Data

Mengxin Zheng
Indiana University Bloomington
zhengme@iu.edu

Qian Lou
University of Central Florida
qian.lou@ucf.edu

Lei Jiang
Indiana University Bloomington
jiang60@iu.edu

2023 60th ACM/IEEE Design Automation Conference (DAC) | 979-8-3503-2348-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/DAC56929.2023.10247719

***Abstract*—It is increasingly important to enable privacy-preserving inference for cloud services based on Transformers. Post-quantum cryptographic techniques, e.g., fully homomorphic encryption (FHE), and multi-party computation (MPC), are popular methods to support private Transformer inference. However, existing works still suffer from prohibitively computational and communicational overhead. In this work, we present, Primer, to enable a fast and accurate Transformer over encrypted data for natural language processing tasks. In particular, Primer is constructed by a hybrid cryptographic protocol optimized for attention-based Transformer models, as well as techniques including computation merge and tokens-first ciphertext packing. Comprehensive experiments on encrypted language modeling show that Primer achieves state-of-the-art accuracy and reduces the inference latency by $90.6\% \sim 97.5\%$ over previous methods.**

***Index Terms*—Fully Homomorphic Encryption, Multi-party Computation, Transformer, Cryptographic Protocol, Private Inference**

I. Introduction

Transformer-based, or more broadly attention-based, models show superior performance over previous methods, becoming increasingly popular in natural language processing (NLP) applications [6]. For example, BERT obtains new state-of-the-art results on eleven NLP tasks, including pushing the GLUE score to 80.5% (7.7% absolute improvement), and even proves superior performance compared to human results on the challenging sentence classification tasks. Server-based Transformer service is an effective way for clients to run their computationally expensive and memory-intensive NLP tasks on powerful cloud servers. During a server-based Transformer service, cloud servers require access to clients' language data, thus introducing potential privacy risks. Therefore, to be able to utilize this technology, it is urgently needed to safeguard the confidentiality of users' biomedical, financial, and other sensitive data that are submitted to servers.

Post-quantum cryptographic protocols, e.g., FHE [8], [13] and MPC [2] are popular methods to enable provably confidential computation on encrypted data. We use Figure 1 to show the overview of private transformer inference, where the client receives cloud services based on Transformer models by only uploading encrypted data generated by cryptographic protocols such as FHE or MPC. This Transformer inference is provably privacy-preserving since data is not revealed to other parties [4], [12]. However, existing works for private Transformer inference based on FHE, e.g., THE-X [4], suffer from enormous latency. For example, THE-X takes more than 3 orders of magnitudes latency than regular Transformer inference. And polynomial approximation of activation in THE-X significantly reduces accuracy, e.g., $< 77\%$ GLUE score ($\sim 8\%$ absolute accuracy decrease).

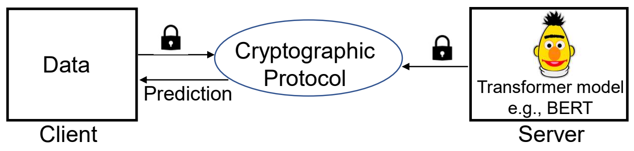

Fig. 1. Overview of private Transformer inference based on cryptographic protocol, e.g., FHE and MPC. The lock represents that data is encrypted.

We identify several challenges to design private Transformer inference, such as the large one-hot word embeddings, complex attention, frequent $SoftMax$, and very deep blocks. Specifically, BERT [6] uses WordPiece embeddings [23] with 30522 token vocabulary and 768 embedding dimensions so that n tokens require n times of 30522×768 matrix-vector multiplication. Directly applying existing techniques to design privacy-preserving embeddings suffers from enormous latency overhead. In addition, we identify that the attention scheme in Transformer models requires massive ciphertext-ciphertext multiplications that cannot directly be implemented by previous methods that are optimized for ciphertext-plaintext multiplications. Moreover, deeper Transformer architecture adds expensive FHE rotations and communicational interactions.

In this work, we present a fast and accurate Transformer inference method, denoted by Primer, over encrypted data. We propose several techniques to construct Primer. In particular, a hybrid cryptographic protocol is proposed to construct a private Transformer, where FHE is used for polynomial operations and MPC is for non-polynomial operations. We call our Primer with this protocol Primer-base. Primer-base is accurate since it removes the polynomial approximation in previous works based on FHE. To reduce the online time of Primer-base, we propose a new hybrid protocol, denoted by HGS, to pre-process most FHE operations. Offloading computations into the offline phase from the online phase is important since offline computations can be computed in advance before inference. We further propose FHGS, denoted by Primer-F to improve the compatibility of HGS on attention computations in Transformer models. Other techniques including computation merge (combined FHGS) and tokens-first packing are presented to further reduce the inference latency. Comprehensive experiments on encrypted language modeling show that Primer achieves state-of-the-art accuracy and reduces the inference latency by $90.6\% \sim 97.5\%$ over previous methods.

II. Background and Motivation

Threat Model. We use Figure 1 to show the overview of our threat model, where servers and clients are semi-honest, e.g., a semi-honest cloud server that attempts to infer clients' input information but follows the cryptographic protocol. Our threat model follows previous work THE-X [4] and Gazelle [10]. The security level of our method is 128 bits for a fair comparison.

Transformer-based Models for NLP Tasks. Transformer-based models [21] achieve state-of-the-art performance in many NLP tasks. A Transformer architecture mainly includes embeddings, stacked encoders, and decoders using Multi-Head Self-Attention (MHSA) and point-wise, fully connected (FC) layers. A model with only encoders, e.g. BERT [6], can be used in discriminative NLP tasks including classification and regression, etc. Meanwhile, a model with decoders, e.g. GPT-2 [17], works for generative NLP tasks including Language Modeling (LM) and machine translation. In particular, embeddings include word embedding and positional embedding. Word embedding converts the input tokens and output tokens (each token is a one-hot vector with a length of d_{oh}) to vectors of dimension d_{emb} by a linear projection. Positional embedding ensures the Transformer model has the sequence order information by adding "positional encodings" λ to the previous word embedding. The embedded representations are fed into MHSA. In MHSA, embedded representations are firstly converted into three categories, key X_K, query X_Q, and value X_V, by linear projections with key weight W_K, query weight W_Q, and value weight W_V, respectively. Then, the output of MHSA is calculated as a weighted sum of the values X_V by $Attention(X_Q, X_K, X_V) = SoftMax(\frac{X_Q X_K^T}{\sqrt{n}})X_V$, where $SoftMax(\frac{X_Q X_K^T}{\sqrt{n}})$ is the weight assigned to each value, and n is token numbers. Instead of only computing the attention once, the multi-head mechanism computes attention multiple times in parallel, and these multiple attentions are simply concatenated and linearly transformed into the expected dimensions as $MultiHead(X_Q, X_K, X_V) = [head_1, ..., head_H]W_O$, where $head_i = Attention(X_Q W_Q^i, X_K W_K^i, X_V W_V^i)$, W_O is a linear projection weight matrix.

Interactive hybrid cryptographic protocol. FHE [8] is an encryption method that enables one to perform computations on encrypted data without decryption. Garbled Circuit (GC) [2], [7] and Secret Sharing (SS) [9] are two paramount methods of multi-party secure computations. An Interactive hybrid cryptographic protocol [10] is proposed to combine the advantages of FHE, GC, and SS. In particular, FHE has superior performance over GC on linear operations, e.g., matrix-vector multiplication. This is because FHE with a ciphertext packing technique supports efficient operations in a SIMD (single instruction multiple data) manner. Therefore, FHE is used to support private linear operations where a client encrypts input and sends it to the server, and the server returns encrypted output to the client that decrypts the received output. In the state-of-the-art mixed protocols [10], [14], [15], GC shows superior performance over HE in non-linear operations such as activation functions. And SS is used to combine GC and HE in the mixed protocol. Inspired by Beaver's Triple [1], FHE can be used to efficiently perform multiplications on two additive secret shares. In this work, we use this interactive hybrid method to construct Primer-base, which is a starting point for our optimization techniques.

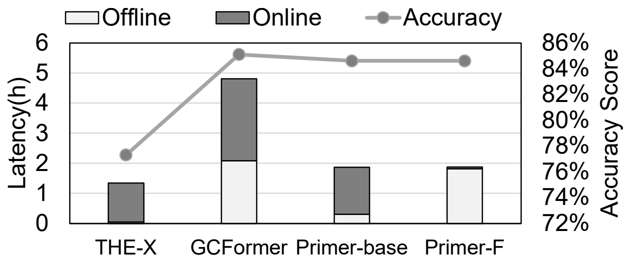

Fig. 2. The latency and accuracy comparisons of prior works, e.g., THE-X and GCFormer, and our work Primer-base and Primer-F on MNLI-m dataset with BERT-base.

Motivation. As Figure 2 shows, prior works like THE-X [4] using only FHE for private inference suffer from low accuracy and enormous online latency due to polynomial approximation and expensive FHE operations. We use prior GC-based work [19] to implement a GCFormer (we convert the Transformer model into a circuit based on binary gates so that GC [2] can implement it). GCFormer achieves an accurate performance, i.e., 85.1% accuracy, but it takes a larger latency than THE-X. Thus, the FHE-based method or GC-based method cannot achieve a low-latency and accurate private Transformer inference. Instead, we follow the interactive and hybrid cryptographic protocol [10] and construct our Primer-base by using GC for non-polynomial operations, FHE for polynomial operations, and SS for secure communication between multiple parties. Primer-base significantly improves the accuracy of THE-X, e.g., 7.3% accuracy increase, and reduces the latency of GCFormer. However, Primer-base still suffers from enormous online latency. This motivates us to propose techniques like the FHGS protocol, denoted by Primer-F, to offload the online computation to the offline phase where computations can be computed before inference. Considering Primer-F still has a large total latency, we have motivations to propose techniques including computation merge, i.e., combined FHGS, and tokens-first ciphertext packing techniques. More details about Primer and related techniques are introduced in the following section III.

III. PRIMER

A. Primer-base Construction

As Figure 3(a) shows, a Transformer-based model involves computations of ❶-❷ embeddings, ❸ derivation of X_Q, X_K and X_V, ❹ scaled dot-product of X_Q and X_K^T, ❺ $SoftMax$, ❻ attention values, and the other linear operations like Fully Connected (FC) computations. Embedding output $X[1]$ is computed by $X[0] \times W_E \times \delta + \lambda$, where $X[0]$, W_E, δ, and λ are the input matrix, embedding weight matrix, positional coefficients, and positional biases, respectively. The embedding output $X[1]$ is multiplied with query weight W_Q, key weight W_K, and value weight W_V to generate query X_Q, key X_K, and value X_V. Multi-Head Self-Attention requires multiple computations of X_Q, key X_K, and value X_V with various weight matrices in parallel. Then a Transformer needs to compute the dot products between the query X_Q and all keys X_K, divide each by $\eta = \sqrt{n}$, apply a $SoftMax$ function to

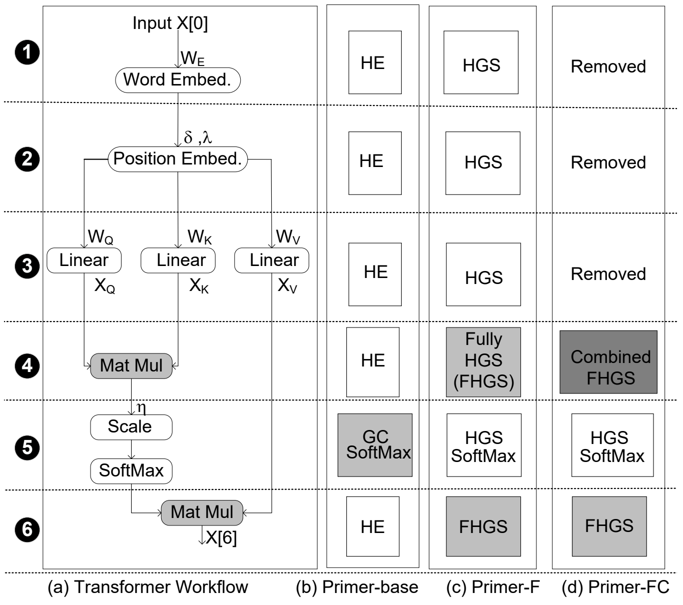

Fig. 3. Private transformer block inference under various Primer protocols.

obtain the weights on the values, and multiply the attention weights with the value X_V. Figure 3(b) illustrates how to construct a basic private Transformer, i.e., Primer-base, using the prior interactive hybrid cryptographic protocol, i.e., FHE (we denote it as HE in Figure 3(b)) is used for polynomial operations in all the steps other than non-polynomial operations, e.g., $SoftMax$. Instead, GC is used for non-polynomial operations.

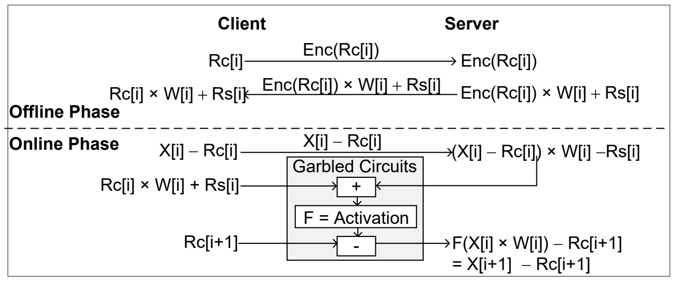

Fig. 4. Our HGS protocol for Transformer's attention operations.

B. Primer-F Construction

Primer-base significantly improves the accuracy of FHE-based methods like THE-X [4] and reduces the latency of MPC-based methods, like GCFormer [19]. However, Primer-base still suffers from enormous online latency. This motivates us to propose techniques like the HGS and FHGS protocol, denoted by Primer-F, to offload the online computation to the offline phase where computations can be computed before inference.

The HGS protocol. Figure 4 shows the mixed HGS protocol. The offline phase in the HGS protocol is used to prepare data for the subsequent online phase. For the i-th layer of a Transformer model, a client first samples a random matrix $Rc[i]$ that is required to have the same size with private input $X[i]$, and then submits the ciphertext Enc($Rc[i]$) to the server for the subsequent multiplication between Enc($Rc[i]$) and the i-th layer weights $W[i]$. A random matrix $Rs[i]$ is generated by the server and $Enc(Rs[i] + Rc[i] \times W[i])$ is sent back to the client. The client performs decryption to get $Rs[i] + Rc[i] \times W[i]$. $Rs[i] + Rc[i] \times W[i]$ held by the client, and $Rs[i]$ held by the server are secret shares of $Rc[i] \times W[i]$. Meanwhile, the offline phase, e.g. garbling, of GC is performed. During the online phase, the difference of $X[i]$ and $Rc[i]$, instead of $X[i]$, is sent to the server. The computation of $(X[i] - Rc[i]) \times W[i] - Rs[i]$ and previous offline computation make the client and server have the additive secret shares of $X[i] \times W[i]$. In this way, the heavy encrypted HE operations of privacy-preserving matrix multiplication of $X[i] \times W[i]$ is calculated offline, and the online overhead is almost removed since only unencrypted computations exist. Then GC is used to perform the subsequent mapping function F, e.g., $ReLU$ activation. Specifically, the garbled Boolean $X[i] \times W[i]$ is derived by the modular sum of secret shares of $X[i] \times W[i]$, then the Boolean circuits of mapping function F are calculated. Finally, a modular subtraction between function F's result and a new random matrix $Rc[i+1]$ is performed to generate secret shares of function F's result. A modular operation circuit is implemented by an adder and a multiplexer [10], [16].

We encapsulate HGS protocol shown in Figure 4 into a module that takes random matrices $Rc[i]$, $Rc[i+1]$, i-th layer input $X[i]$, weight matrix $W[i]$ as inputs, and generates $X[i+1] - Rc[i+1] = F(X[i] \times W[i]) - Rc[i+1]$. Here $F()$ function can be an identity function or an activation function. The i-th layer input $X[i]$ can be removed if the server holds $X[i] - Rc[i]$. $W[0] = W_E, W[2] = \sigma = 1$, and λ is added to $X[1]$ instead of multiplication. As Figure 3(c) shows, steps in the Transformer including ❶ ❷, ❸, ❺, and the other FC computations can be performed by the HGS protocol; however, steps ❹ and ❻ cannot be directly constructed by the additive HGS protocol since HGS only supports additive computations including ciphertext additions and ciphertext-plaintext multiplication. This is because the HGS protocol that depends on an additive HE scheme is only sufficient for modules where weights are always not encrypted. Therefore, HGS cannot transform ciphertext-ciphertext multiplications in steps ❹ and ❻ on secret shares into the offline phase.

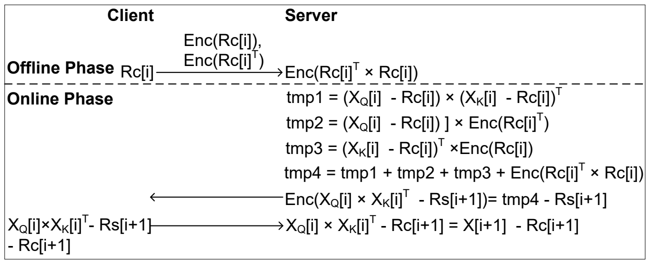
Client
Server
Enc(Rc[i]),
Enc(Rc[i]^T)
Offline Phase
Rc[i]
Enc(Rc[i]^T × Rc[i])
Online Phase
tmp1 = (X_Q[i] - Rc[i]) × (X_K[i] - Rc[i])^T
tmp2 = (X_Q[i] - Rc[i])] × Enc(Rc[i]^T)
tmp3 = (X_K[i] - Rc[i])^T ×Enc(Rc[i])
tmp4 = tmp1 + tmp2 + tmp3 + Enc(Rc[i]^T × Rc[i])
Enc(X_Q[i] × X_K[i]^T - Rs[i+1])= tmp4 - Rs[i+1]
X_Q[i]×X_K[i]^T- Rs[i+1] - Rc[i+1]
X_Q[i] × X_K[i]^T - Rc[i+1] = X[i+1] - Rc[i+1]

Fig. 5. Our Fully HGS (FHGS) protocol for attention operations.

The Fully HGS (FHGS) protocol for ciphertext-ciphertext operations. The step ❹ of attention in Transformer is different from steps ❷ through ❸ where weights are not encrypted and only inputs are encrypted so that private inference is a type of ciphertext-plaintext operations. In step ❹, however, all query, key, and value matrices are encrypted. The Attention operations require ciphertext-ciphertext operations. Ciphertext-ciphertext operations are not only more expensive than ciphertext-plaintext operations but also cannot directly use HGS method, thus we propose the FHGS protocol to solve this problem.

Inspired by Beaver's Triple method [1], we propose a Fully HGS (FHGS) protocol to empower the prior additive HGS to efficiently support ciphertext-ciphertext operations such as $X_Q \times X_K^T$ in Transformer models. Figure 5 shows our FHGS protocol for step ❹ $X_Q[i] \times X_K[i]^T$. Since $X_Q[i]$ and $X_K[i]^T$ are both ciphertexts, additive HGS cannot offload $X_Q[i] \times X_K[i]^T$ operations. FHGS pre-computes encrypted triples including $Enc(Rc[i])$, $Enc(Rc[i]^T)$, and $Enc(Rc[i]^T \times Rc[i])$ for the usage of the subsequent online process. During the online phase in FHGS, the server has access to $X_Q[i] - Rc[i]$ and $(X_K[i] - Rc[i])^T$ although $X_Q[i]$ and $X_K[i]^T$ are not seen by the server. So an important intermediate result $tmp1 = (X_Q[i] - Rc[i]) \times (X_K[i] - Rc[i])^T$ can be derived. The key idea to obtain our target $tmp4$, a ciphertext of $X_Q[i] \times X_K[i]^T$, is that it can be calculated by subtracting three entries from $tmp1$, where this subtraction can be done by $tmp4 = tmp1 + tmp2 + tmp3 + Enc(Rc[i]^T \times Rc[i])$. In order to preserve the privacy of $X_Q[i] \times X_K[i]^T$, the server transmits its additive secret sharing ciphertext $Enc(X_Q[i] \times X_K[i]^T - Rs[i+1])$, instead of $Enc(X_Q[i] \times X_K[i]^T)$, to the client who can decrypt $Enc(X_Q[i] \times X_K[i]^T - Rs[i+1])$ and obtain $X_Q[i] \times X_K[i]^T - Rs[i+1]$. In this way, the client and the server acquire additive secret shares of $X_Q[i] \times X_K[i]^T$. Optionally, the client can further share $X_Q[i] \times X_K[i]^T - Rs[i+1] - Rc[i+1]$ with the server to obtain new secret shares of $X_Q[i] \times X_K[i]^T$. At last, the FHGS protocol is enclosed into a module that takes random matrices $Rc[i]$, $Rc[i+1]$, $Rs[i+1]$, $X_Q[i] - Rc[i]$, and $(X_K[i] - Rc[i])^T$ as inputs, and outputs the secret shares of $X_Q[i] \times X_K[i]^T$.

Privacy analysis. The $X_Q[i]$, $X_K[i]$, $X_Q[i] \times X_K[i]^T$ are confidential to both the client and the server, which ensures our FHGS protocol is privacy-preserving. The server that has no access to HE private key cannot decrypt ciphertexts including $Enc(Rc[i]^T \times Rc[i])$, $tmp1$, $tmp2$, $tmp3$, $tmp4$, and $tmp4 - Rs[i+1]$. Only secret shares of $X_Q[i]$, $X_K[i]$, $X_Q[i] \times X_K[i]^T$ can be accessed by client and server. Also, FHGS completely offloads complex and expensive ciphertext-ciphertext operations from the online phase into the offline phase since $Rc[i]$ and $Rc[i]^T$ are pre-sampled and their product can be calculated in advance, which enables a additive HE scheme to efficiently perform privacy-preserving ciphertext-ciphertext Transformer operations.

C. Primer-FC by Combined FHGS (CHGS)

We further reduce the computational and communicational overhead of previous technologies by a combined FHGS (CHGS) method that can combine adjacent HGS layers. The CHGS processes multiple stacked operations using a single calculation, and most HE-based operations are moved to the offline phase. As Figure 3(d) shows, our CHGS module removes its previous HGS operations by incorporating three HGS modules into the adjacent FHGS module. The key idea is that the combined target is $(X[i] \times W_E + \lambda) \times W_Q \times [(X[i] \times W_E + \lambda) \times W_K]^T = X_Q[i] \times X_K[i]^T$ which can be derived from $tmp1 = ((X[i] - Rc[i]) \times W_E + \lambda) \times W_Q \times [((X[i] - Rc[i]) \times W_E + \lambda) \times W_K]^T$ by $tmp1 - tmp2 + tmp3 + tmp4 - tmp5 + tmp6 - tmp7 = result$. The combined weight W_M, $tmp6$, and $tmp7$ can be calculated in the offline phase. The server sends $result\text{-}Rs[i+1]$ to the client so that the client obtains the decryption of $result\text{-}Rs[i+1]$ and the server has $Rs[i+1]$. The client can also subtract the decryption of $result\text{-}Rs[i+1]$ with $Rc[i+1]$ to construct a new secret sharing. Using CHGS, 4-time interactions in Figure 3(d) can be reduced into 1-time interaction. The improvement details of CHGS are discussed in the following results section.

Algorithm: Encrypted HE matrix multiplications by feature-based packing and our tokens-first packing
Input: X[i], W[i]
Output: Enc(X[i+1]) = Enc(X[i])×W[i]

```
1. c=0,s=0
2. for (h=0; h<n; h++):
3.     for (j=0; j<$d_{oh}$; j++):
4.         p[c][s] = X[i][h][j];
5.         s++;
6.         if (s==M or j==$d_{oh}$):
7.             s=0; c++;
8.             cipher[c]=Enc(p[c]);
9. for (j=0; j<c; j++):
10.    for (i=0; i<M; i++):
11.        tmpC=Rotate(cipher[j],i);
12.        for (g=0; g< $d_{emb}$; g++):
13.            result[j][g] +=
               tmpC×$W_R$[i][g];
14.    Merge and Add Ciphers.
```

(a) Features-based packing

```
1. c=0,s=0
2. for (j=0; j<$d_{oh}$; j++):
3.     for (h=0; h<n; h++):
4.         p[c][s] = X[i][h][j];
5.         s++;
6.         if (h==0 and (s+n)<=M):
7.             s=0; c++;
8.             cipher[c]=Enc(p[c]);
9. for (j=0; j<c; j++):
10.    for (i=0; i<M; i=i+n):
11.        tmpC=Rotate(cipher[j],i);
12.        for (g=0; g< $d_{emb}$; g++):
13.            result[j][g] +=
               tmpC×$W_R$[i][g];
14.    Merge and Add Ciphers.
```

(b) Tokens-first packing

Fig. 6. The comparison of features-based packing and our tokens-first packing.

D. Tokens-first Packing, i.e., Primer-FPC Construction

Embedding is used to compress a large and sparse one-hot vector into a small and dense vector. How to efficiently support high-dimension ($> 30K$) matrix multiplication is not studied. Each word (token) will be a vector of size 30522 which is larger than the ciphertext slot numbers. For multiple words in a sentence, how to pack these words into ciphertext is a new challenge. We propose tokens-first packing to tackle this challenge, instead of prior feature-based ciphertext packing used in [3], [5], [10], [16]. In feature-based ciphertext batching, multiple features (e.g. pixels in an input image) are batched into the same ciphertext. However, we found that directly applying the feature-based packing method on Transformer-based NLP models introduces massive FHE rotations.

We propose a tokens-first packing method instead of the prior features-based packing method to reduce the homomorphic rotations in Primer-FC. Figure 6(a) depicts the pseudo-code of encrypted matrix multiplication based on feature-base packing between $X[i]$, i.e., $X[i][0:n][0:d_{oh}]$ and $W[i]$, i.e., $W[i][0:d_{oh}][0:d_{emb}]$, where n, d_{oh}, and d_{emb} are input tokens number, one-hot dimension of a token, and embedding dimension, respectively. Here $X[i][0:n][0:d_{oh}]$ represents the shape of matrix $X[i]$. The result of matrix multiplication is $X[i+1]$ with a size of $X[i+1][n][d_{emb}]$. Lines 2 through 8 are used to pack the input matrix $X[i][0:n][0:d_{oh}]$ into c plaintexts $p[0:c]$ and encrypt them into c ciphertexts via the encryption function $Enc()$. Each plaintext p has M slots so it can hold M entries. In the features-based packing, the one-hot features $X[i][h][0:j]$ of a token h are first placed into plaintexts, then the features of the next token $h+1$ are packed until all tokens' features are packed and encrypted. The features-based packing requires $c \times M$ $Rotate$ operations since each ciphertext with M features needs M rotations shown in line 9 $\sim$ line 14. One key observation is that features from different tokens are independent and they are not required to accumulate in the matrix multiplication, which motivates us to propose tokens-first packing to batch tokens as much as possible into the same ciphertext. As Figure 6(b) shows, the lines of 2, 3, 6, and 10 of features-based packing are replaced so that the j-th feature $X[i][0:n][j]$ of all n tokens are packed into a ciphertext, then the $j+1$-th feature will be packed. Using this tokens-first packing, one ciphertext only has $\sim \frac{M}{n}$ features so that one ciphertext only requires $\sim \frac{M}{n}$ $Rotate$ operations. Considering both features-based packing and our tokens-first packing have similar ciphertext numbers c, our tokens-first packing reduces $c \times (M - \frac{M}{n})$ rotations.

TABLE I
COMPARISON OF OUR METHOD AND OTHER WORKS ON PRIVATE BERT-BASE INFERENCE. "/" MEANS NON-APPLICABLE.

Scheme	Offline(s)	Online(s)	Total(s)	Acc.(%)
THE-X	/	$4.7K$	$4.7K$	77.3%
GCFormer	$7.5K$	$9.8K$	$17.3K$	85.1%
Primer-F	$6.5K$	$0.04K$	$6.54K$	84.6%
Primer-FPC (Primer)	$0.4K$	$0.04K$	$0.44K$	84.6%

IV. Experimental Methodology

System setup and security analysis. We run the privacy-preserving Transformer experiments on two instances that are equipped with an Intel Xeon E7-4850 CPU and 128 GB DRAM, and each instance was provided with 4 threads. In our current system setup, the average network delay between these two instances is 2.3 ms and the bandwidth is about 100 MB/s. The layer-wise PAHE used in Primer is implemented by SEAL [20] libraries where only additive HE operations and rotations are used and ciphertext-ciphertext multiplications are not required. We adopt an extension version of JustGarble tool [2] used in [10] to implement GC-based operations, including additions of secret sharings and activation functions. The HE parameters and GC settings are selected to provide 128-bit security level. The inputs and weights use 15-bit fix-point representation and the intermediate results are truncated into 15 bits to avoid overflow. The training, fine-tuning, and testing of the Transformer on plaintext was implemented in Python on 4 NVIDIA Tesla V100 GPUs.

Transformer architecture and NLP datasets. We evaluated Primer on five discriminative NLP models shown in Table III: BERT-Tiny, BERT-small, BERT-base, BERT-medium, BERT-large. The hyper-parameters of these models are listed in Table III. For example, the BERT-tiny model has $N=3$ blocks, $d_{emb}=768$ embedding dimensions, $H=12$ attention heads, and $n=30$ input tokens. Datasets for five BERT tasks are SQuAD1 [18], SQuAD2 [11], and MNLI-m, MRPC, SST-2 from GLUE benchmarks [22].

V. Results and Analysis

Comparison with Prior Works. We compare our primer with prior works on private BERT-base inference for MNLI-m dataset in table I. Prior work, e.g., THE-X [4] that only uses FHE for private inference only achieves 77.3% accuracy with $4.7k$ seconds latency due to polynomial approximation and expensive FHE operations. We use prior GC-based work [19] to implement a GCFormer. It achieves an accurate performance, i.e., 85.1% accuracy, but it takes a larger latency than THE-X. Our Primer-F significantly improves the accuracy of THE-X, e.g., 7.3% accuracy increase, and reduces the latency of GCFormer. To reduce the large offline latency of Primer-F, we further propose Primer-FPC, i.e., Primer, with tokens-first packing and combined FHGS. Our primer only takes $\sim 0.4k$-second latency, thus achieving a $\sim 16\times$ latency reduction.

Ablation Study. Table II describes the performance breakdown and the ablation effects of proposed techniques using BERT-base model with $n=30$ on MNLI-m dataset. The Primer-base implemented by FHE and MPC protocols requires $\sim$ 6553 seconds latency to perform one inference on a sentence in MNLI-m dataset and achieves 84.6% accuracy. We further propose FHGS, denoted by Primer-F to Offload computations into the offline phase from the online phase, which significantly shrinks the offline latency from $\sim$ 6553 seconds to $\sim$ 41 seconds, introducing almost $160\times$ latency reduction. Primer-FP is proposed to reduce the latency of embedding layers and the following layers that include HE operations. It further decreases 5.3% online latency over Primer-F and has $16\times$ offline latency reduction. Primer-FPC has the similar offline latency and accuracy with Primer-topk, but further reduces the online latency by 9.2%. Table II shows Primer (Primer-FPC) achieves competitive NLP accuracy and reduces the online and offline inference latency by $90.6\% \sim 97.5\%$ over Primer-base.

Results on Different Models. Table III studies the effects of different BERT models and datasets on Primer. Privacy-preserving BERT-tiny with only 3 Transformer blocks achieves average 81.7% accuracy on three GLUE datasets including MNLI-m , MRPC, SST-2, and average 81.4% F1 test accuracy on SQuAD1 and SQuAD2. Primer with BERT-tiny requires 10.6 seconds to perform an inference or classification on a sequence with 30 tokens, thereby attaining a throughput of 2.83 tokens per second. Also, the communicational message size between clients and a server is $\sim$ 0.9 GB. Primer with BERT-small or BERT-base achieves $2.4\% \sim 7.5\%$ higher accuracy by adding more Transformer blocks, but increases latency by $78.3\% \sim 230\%$. Also, Primer with BERT-small and BERT-base require more than $2\times$ and $3\times$ message size, respectively, than BERT-tiny based Primer. BERT-medium with larger embedding dimension and BERT-large with 24 block numbers are also supported by Primer, and they take 45.1 seconds and 91.6

TABLE II
A PERFORMANCE ABLATION STUDY OF PROPOSED TECHNIQUES. PRIMER-BASE IS CONSTRUCTED BY A HYBRID CRYPTOGRAPHIC PROTOCOL, WHERE FHE IS USED FOR POLYNOMIAL OPERATIONS AND MPC IS FOR NON-POLYNOMIAL OPERATIONS. PRIMER-F, PRIMER-FP, AND PRIMER-FPC APPLY FHGS PROTOCOLS, TOKENS-FIRST PACKING, AND CHGS TECHNIQUES, IN A CASCADE MANNER. THE OFFLINE AND ONLINE LATENCY (SECONDS) OF EACH STEP IN BERT-BASE ARE LISTED. ACC. MEANS ACCURACY ON THE MNLI-M DATASET. "/" MEANS NON-APPLICABLE.

Scheme	❶❷ $Embed$		❸ QKV		❹ $Q \times K$		❺ $SoftMax$		❻ $Atten.Value$		Others		Total		Acc.(%)
	offline	online	offline	online	offline	online	offline	online	offline	online	offline	online	offline	online	
Primer-base	/	3094.4	/	190.6	/	25	0.01	16.4	0.8	2.3	/	3224.5	0.81	6553.2	84.6
+FHGS (Primer-F)	3094.4	0.8	190.6	3.9	14	9.9	0.01	16.4	0.8	2.3	3224.5	7.9	6524.3	41.2	84.6
+Pack (Primer-FP)	134.5	0.8	39	3.9	10.5	7.7	0.01	16.4	0.8	2.3	220.4	7.9	405.2	39	84.6
+CHGS (Primer-FPC)	0	0	0	0	178.2	11.6	0.01	14.8	0.8	2.3	220.4	6.7	399.4	35.4	84.6

TABLE III
THE PERFORMANCE OF PRIMER OVER VARIOUS BERT MODELS ON MULTIPLE DATASETS.

Model	Hyper-parameters				GLUE and SQuAD Accuracy (%)					Latency(s)		Throughput	Message
	N	d_{emb}	H	n	MNLI-m	MRPC	SST-2	SQuAD1	SQuAD2	offline	online	tokens/s	GB
BERT-tiny	3	768	12	30	77.6	79.3	88.2	86.2	76.6	318.5	10.6	2.83	0.9
BERT-small	6	768	12	30	81.6	84.5	91.1	88.5	78.2	345.2	18.9	1.59	1.8
BERT-base	12	768	12	30	84.6	86.3	92.5	90.7	80.3	399.4	35.4	0.85	3.6
BERT-medium	12	1024	16	30	85.4	86.3	93.1	92.2	81.6	452.8	45.1	0.67	3.9
BERT-large	24	1024	16	30	86.6	87.6	93.5	93.1	82.9	586.4	91.6	0.33	7.9

seconds to perform an inference on a sentence with 30 tokens. BERT-large achieves state-of-the-art accuracy on GLUE and SQuAD benchmarks.

VI. CONCLUSION

In this paper, we present Primer to enable a fast and accurate privacy-preserving Transformer for NLP tasks. First, a naïve version of our Primer, called Primer-base, is constructed by a hybrid interactive cryptographic protocol. Secondly, we propose tokens-first packing instead of prior features-first packing to reduce the offline and online overhead brought by HE. Finally, we demonstrate that multiple secret sharing layers in the Transformer can be combined to reduce the latency. Primer establishes a solid baseline and shed the light on private Transformer inference over encrypted data.

ACKNOWLEDGMENT

This work was supported in part by NSF awards CCF-1908992, CCF-1909509, and CCF-2105972.

REFERENCES

[1] D. Beaver, "Precomputing oblivious transfer," in *Annual International Cryptology Conference*, pages 97–109, Springer, 1995.
[2] M. Bellare, *et al.*, "Efficient Garbling from a Fixed-Key Blockcipher," Cryptology ePrint Archive, Report 2013/426, 2013.
[3] A. Brutzkus, *et al.*, "Low latency privacy preserving inference," in *International Conference on Machine Learning*, pages 812–821, 2019.
[4] T. Chen, *et al.*, "THE-X: Privacy-Preserving Transformer Inference with Homomorphic Encryption," in *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3510–3520, Association for Computational Linguistics, Dublin, Ireland, May 2022.
[5] R. Dathathri, *et al.*, "EVA: An Encrypted Vector Arithmetic Language and Compiler for Efficient Homomorphic Computation," in *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI 2020, page 546–561, Association for Computing Machinery, New York, NY, USA, 2020.
[6] J. Devlin, *et al.*, "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: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Association for Computational Linguistics, Minneapolis, Minnesota, June 2019.
[7] B. Feng, *et al.*, "CryptoGRU: Low Latency Privacy-Preserving Text Analysis With GRU," *arXiv preprint arXiv:2010.11796*, 2020.
[8] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 169–178, 2009.
[9] O. Goldreich, *et al.*, "How to Play ANY Mental Game," in *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, page 218–229, Association for Computing Machinery, New York, NY, USA, 1987.
[10] C. Juvekar *et al.*, "GAZELLE: A Low Latency Framework for Secure Neural Network Inference," in *USENIX Security Symposium*, 2018.
[11] G. Lee, *et al.*, "SQuAD2-CR: Semi-supervised Annotation for Cause and Rationales for Unanswerability in SQuAD 2.0," in *Proceedings of the 12th Language Resources and Evaluation Conference*, European Language Resources Association, Marseille, France, May 2020.
[12] D. Li, *et al.*, "MPCFormer: fast, performant and private Transformer inference with MPC," *arXiv preprint arXiv:2211.01452*, 2022.
[13] Q. Lou and L. Jiang, "SHE: A Fast and Accurate Deep Neural Network for Encrypted Data," in *Advances in Neural Information Processing Systems*, pages 10035–10043, 2019.
[14] Q. Lou, *et al.*, "AutoPrivacy: Automated Layer-wise Parameter Selection for Secure Neural Network Inference," in *Advances in Neural Information Processing Systems*, edited by H. Larochelle, *et al.*, volume 33, pages 8638–8647, Curran Associates, Inc., 2020, .
[15] Q. Lou, *et al.*, "SAFENet: A Secure, Accurate and Fast Neural Network Inference," in *International Conference on Learning Representations*, 2021.
[16] P. Mishra, *et al.*, "Delphi: A Cryptographic Inference Service for Neural Networks," in *USENIX Security Symposium*, USENIX Association, Boston, MA, August 2020.
[17] A. Radford, *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, 1(8):9, 2019.
[18] P. Rajpurkar, *et al.*, "SQuAD: 100, 000+ Questions for Machine Comprehension of Text," *CoRR*, abs/1606.05250, 2016.
[19] B. D. Rouhani *et al.*, "DeepSecure: Scalable Provably-Secure Deep Learning," in *ACM/IEEE Design Automation Conference*, 2018.
[20] "Microsoft SEAL (release 4.0)," , March 2022, microsoft Research, Redmond, WA.
[21] A. Vaswani, *et al.*, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, edited by I. Guyon, *et al.*, volume 30, Curran Associates, Inc., 2017.
[22] A. Wang, *et al.*, "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding," in *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Association for Computational Linguistics, Brussels, Belgium, November 2018.
[23] Y. Wu, *et al.*, "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation," *CoRR*, abs/1609.08144, 2016.