基于可信执行环境 TEE 的大模型保护技术

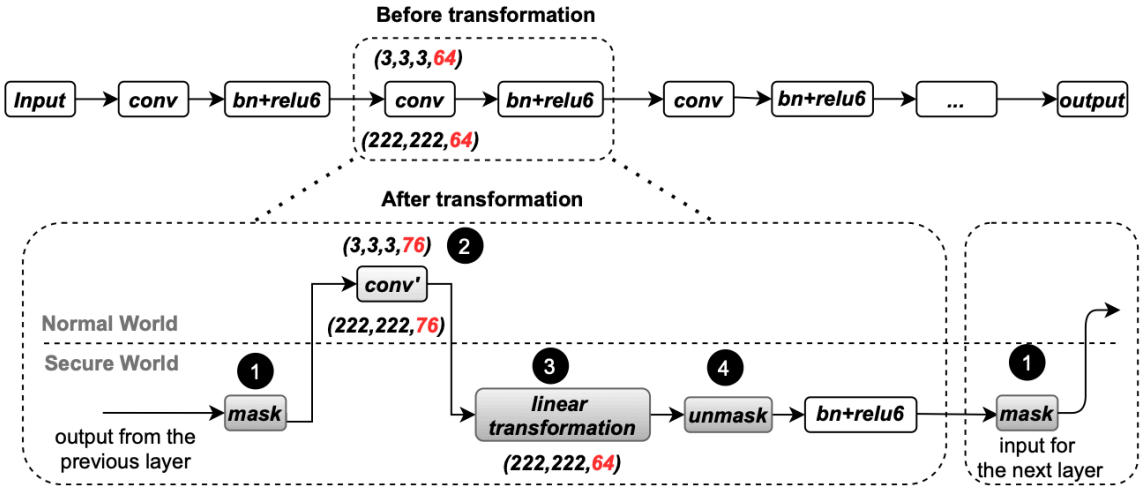
Li Kaihua

Oct 19, 2023

PROBLEM

大模型推理依赖 GPU 计算，而目前 TEE 无法提供 GPU 速计算能力。

ShadowNet 提出一种方法将线性算子混淆后，在不泄漏模型权重的前提下，将线性计算外包给不可信的 GPU 以获得加速。



Transformer

目前大模型主要基于 Transformer 架构构建，如 LLAMA-2-7B 是基于 Transformer Decoder-only 的架构构建。

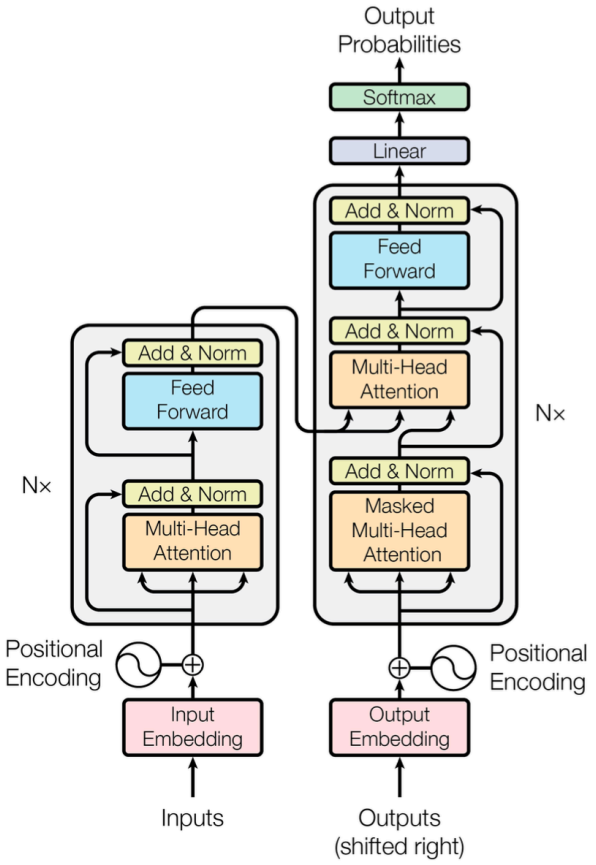


Figure 1: The Transformer - model architecture.

1. Transformer 由若干个 Block 组成。
2. 每个 Block 主要由 Multi-Head Attention 与 FeedForward 构成。
3. Multi-Head Attention 与 FeedForward 都进行 Add 和 Norm 处理。

观察到 TransformerBlock 中 Norm 与 Attention-Softmax 为非线形计算，此外，主要计算形式为线性计算。

LLAMA-2-7B

目前大模型主要基于 Transformer 架构构建，如 LLAMA-2-7B 是基于 Transformer Decoder-only 的架构构建。

```
==========================================================================
Total params: 6,738,415,616
Trainable params: 6,738,415,616
Non-trainable params: 0
==========================================================================
Transformer(
  (tok_embeddings): ParallelEmbedding()
  (layers): ModuleList(
    (0-31): 32 x TransformerBlock(
      (attention): Attention(
        (wq): ColumnParallelLinear()
        (wk): ColumnParallelLinear()
        (wv): ColumnParallelLinear()
        (wo): RowParallelLinear()
      )
      (feed_forward): FeedForward(
        (w1): ColumnParallelLinear()
        (w2): RowParallelLinear()
        (w3): ColumnParallelLinear()
      )
      (attention_norm): RMSNorm()
      (ffn_norm): RMSNorm()
    )
  )
  (norm): RMSNorm()
```

- LLAMA-2-7B 同样采用了 TransformerBlock 结构。
- 每个 TransformerBlock 有 202,383,240 参数。
- TransormerBlock 中 FeedForward 有 135,266,304 参数。
- TransormerBlock 中 Attention 有 67,108,864 参数。
- TransformerBlock 参数占模型参数的 96.10%

LLAMA-2-7B Profiler

Type	CPU%	CPU(s)	CUDA(s)
TransformerBlock	90.12%	7.698s	98.41% / 2.110s
FeedForward	15.71%	1.342s	1.131s
Attention	48.35%	4.130s	775.642ms
Attention-Linear-1	11.70%	1.035s	416.016ms
Attention-RoPE	10.10%	893.678ms	72.283ms
Attention-Linear-2	10.66%	942.944ms	78.705ms
Attention-Softmax	4.64%	395.959ms	36.178ms
Attention-Linear-3	8.90%	787.683ms	189.696ms
NORM	18.81%	1.607s	183.498ms
NORM-1	10.67%	911.261ms	89.713ms
NORM-2	10.24%	874.353ms	91.078ms

观察 LLAMA-2-7B 推理过程，FeedForward 与 Attention 层占 TransformerBlock CUDA 总耗时 2.110s 中的 1.131s 与 0.776s，在利用 CUDA 进行加速的计算中，Attention 与 FeedForward 为加速计算主要内容。

ShadowNet

数学基础:

Linear Transformation. Linear transformation is a function f defined on vector spaces V and T over the same field $\mathbb{F}$, $f : V \to T$. For any two vectors $u, v \in V$ and any scalar $c \in \mathbb{F}$, the following two conditions are satisfied:

$$\begin{aligned} additivity &: f(u+v) = f(u) + f(v) \\ homogeneity &: f(cu) = cf(u) \end{aligned} \tag{5}$$

ShadowNet 针对 Convlution 算子提出安全外包 GPU 计算的方法，

$$\hat{W}^T = W^T \cdot \Lambda + F$$

$$\Rightarrow [\hat{w}_1, \cdots, \hat{w}_n] = [w_1, \cdots, w_n] \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} + [f_1, \cdots, f_n]$$

ShadowNet

Conv 满足线性算子的特性，因此有

$$\begin{aligned}\mathrm{Conv}(X, \hat{W}^T) &= \mathrm{Conv}(X, W^T.\Lambda + F)) \\ &= \mathrm{Conv}(X, W^T) \cdot \Lambda + \mathrm{Conv}(X, F)\end{aligned}$$

通过计算

$$Y = \left(\mathrm{Conv}(X, \hat{W}^T) - \mathrm{Conv}(X, F)\right) \cdot \Lambda^{-1}$$

恢复得到原始结果 Y

ShadowNet

方法分析：

$$Y = \left(\mathrm{Conv}(X, \hat{W}^T) - \mathrm{Conv}(X, F)\right) \cdot \Lambda^{-1}$$

- W、Λ 仅在可信环境可见，$\widehat{W}$ 与 F 在不可信环境中可见。
- 在不可信环境中计算 $\mathrm{Conv}\left(X, \widehat{W}^T\right)$ 与 $\mathrm{Conv}(X, F)$，在可信环境中计算 $\left(\mathrm{Conv}\left(X, \widehat{W}^T\right) - \mathrm{Conv}(X, F)\right) \cdot \Lambda^{-1}$
- 引入一倍的计算开销，可以通过复用 F 减少计算开销，但会降低安全性。
- 适用于 Convlution 层中 W 参数数量远小于 X 的特点。
- Y 与 Y' 存在关系 $Y = Y'\Lambda^{-1}$，不够安全。

ShadowNet

MASK & SHUFFLE

MASK

有随机矩阵 M

$$X' = X + M$$

SHUFFLE

有重排矩阵 P, 满足

$P[i][j] = 1,$

$P[i][:j] = 0 \& P[i][j+1:] = 0 \& P[:i][j] = 0 \& P[i+1:][j] = 0$

$$X'' = X'P$$
$$\widehat{W}' = \widehat{W}P$$
$$Y'' = \mathrm{Conv}(X'', W'), F' = \mathrm{Conv}(X'', F)$$

ShadowNet

MASK & SHUFFLE

UNSHUFFLE

$$Y' = (Y'' - F')\Lambda^{-1}P^{-1}$$

UNMASK

$$Y = Y' - \mathrm{Conv}(X, M)$$

方法分析

- SHUFFLE 通过重排 X 上的特征顺序，使潜在攻击者丢失 Y'' 与 Y' 之间的映射($Y'' = Y'\Lambda$)的观察。
- MASK 通过在 X 上添加随机矩阵 M，使得潜在攻击者丢失 X 到 Y'' 之间的映射观察。

ShadowNet For LLAMA-2-7B

KEY PROBLEMs

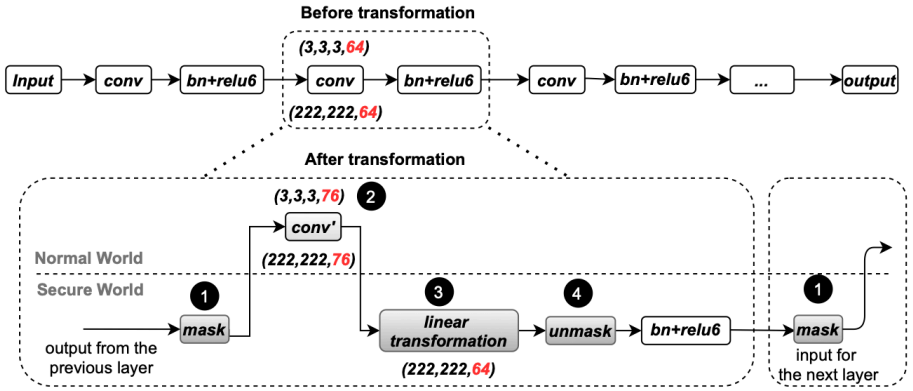
1. ShadowNet 方法原理能否适用于 LLAMA-2-7B(TransformerBlock)?
   - ShadowNet 方法适用于线性算子，Attention 与 FeedForward 存在大量线形计算，混淆权重进行安全外包 GPU 计算在原理上可行。
2. 使用 ShadowNet 方法会为 LLAMA-2-7B 带来多大的开销?
   - Mask & Shuffle 操作需要将 X 移动到 TEE 中，其带来的额外开销为 $O(Size(X)) * OP(TEE\_MEM\_MOVE+MASK+SHUFFLE)$。
   - Weight Transformation 操作，其转换的额外开销为 $O(Size(W) + Szie(X)) * OP(TEE\_MEM\_MOVE+TRANSFORMATION)$。

如何减少引入 ShadowNet 方法所带来的额外开销?

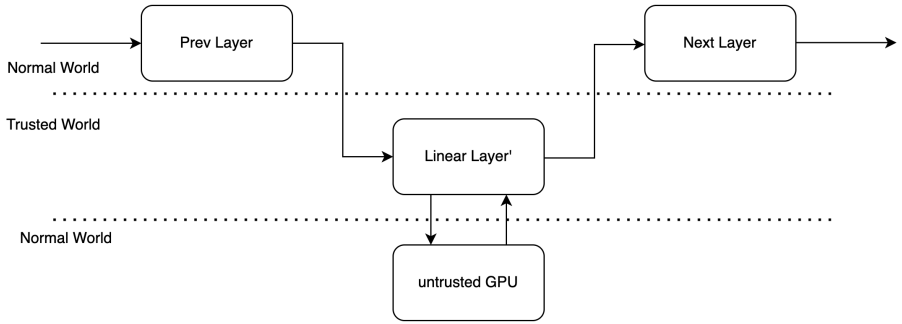
ShadowNet For LLAMA-2-7B

如何减少引入 ShadowNet 方法所带来的额外开销?

翻转主要计算域，保护关键网络层



ShadowNet 面向嵌入式设备的轻量级模型(如 MobileNet)，其主要计算域在 TEE，将部分 Conv 计算通过混淆权重方法安全地外包给不可信 GPU。



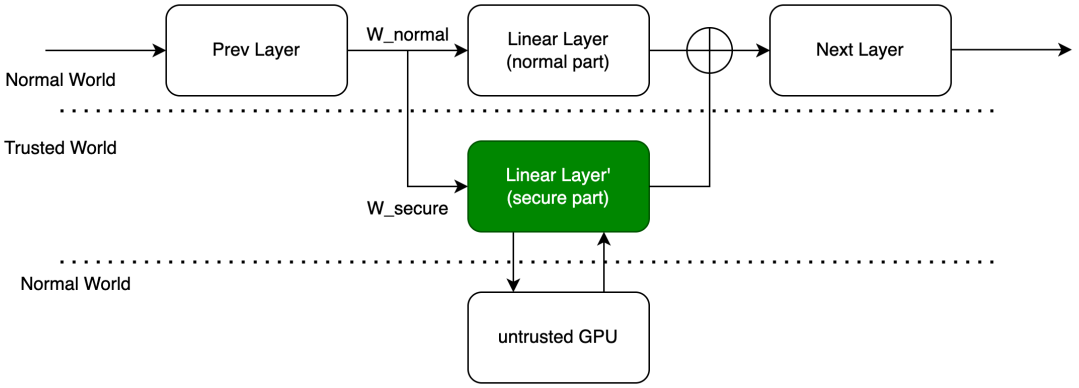
LLAMA-2-7B 等大模型的保护目标主要是参数 W，对输入 X 与输出 Y 并不敏感，因此可以考虑翻转计算域，仅将关键网络层迁移至 TEE 中以保证计算过程的安全性。

ShadowNet For LLAMA-2-7B

如何减少引入 ShadowNet 方法所带来的额外开销?

垂直切割数据域，安全与性能权衡

在 LLAMA-2-7B 中，仅 FeedForward 一层就有 135,266,304 参数，若以 FP32 存储参数，则需要约 541.1 MB 内存空间，若保护所有共 32 层 FeedForward 参数，则需要约 16.9 GB 空间，对于内存资源有限的 TEE 而言是不可行的。

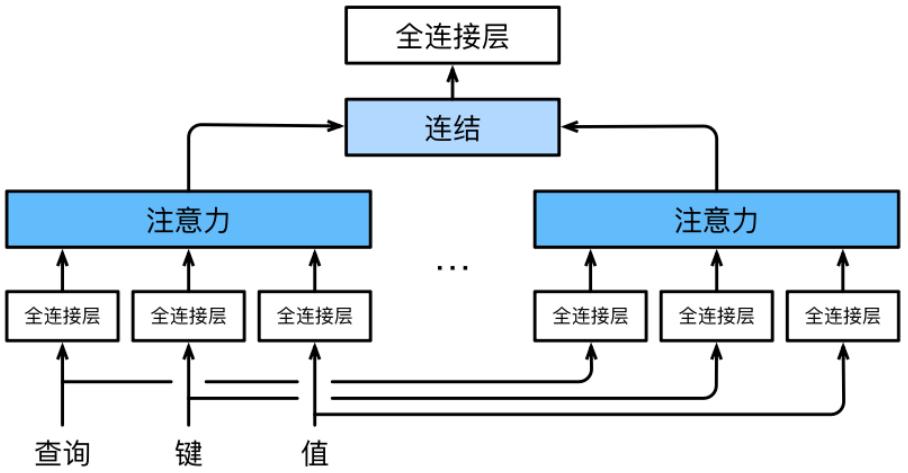


通过垂直切分数据域，保护部分参数 W 以减少计算和存储开销。

ShadowNet For LLAMA-2-7B

如何减少引入 ShadowNet 方法所带来的额外开销?

流水线并行技术，TEE/HOST 协同计算



Multi-Head Attention 由多个平行的 Attention 构成，而每个 Attention 都包括 Q、K、V 的线性投影，$\text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$ 两个流程。最后连接所有 Attention 并通过全连接层输出。

Q、K、V 的线性投影包含可学习参数，需要依赖于 TEE 进行保护，而 $\text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$ 不存在可学习参数，可以在不可信环境直接计算，即表明 TEE 与 HOST 计算存在重叠区间，因此可以考虑若干 Attention 的计算流程间构成流水线，从而协同 TEE 与 HOST 之间的计算。