计 算 机 研 究 与 发 展
Journal of Computer Research and Development
DOI: 10.7544/issn1000-1239.202330966
61(5): 1218−1229, 2024

一种基于安全多方计算的快速 Transformer 安全推理方案

刘伟欣[1] 管晔玮[1] 霍嘉荣[1] 丁元朝[1] 郭 华[1,2] 李 博[2]
[1](北京航空航天大学网络空间安全学院 北京 100191)
[2](复杂关键软件环境全国重点实验室(北京航空航天大学) 北京 100878)
(sy2339130lwx@buaa.edu.cn)

A Fast and Secure Transformer Inference Scheme with Secure Multi-Party Computation

Liu Weixin[1], Guan Yewei[1], Huo Jiarong[1], Ding Yuanchao[1], Guo Hua[1,2], and Li Bo[2]
[1](*School of Cyber Science and Technology*, *Beihang University*, *Beijing* 100191)
[2](*State Key Laboratory of Complex & Critical Software Environment* (*Beihang University*), *Beijing* 100878)

Abstract Transformer has been widely used in many fields such as natural language processing and computer vision, and has outstanding performance. The users' data will be leaked to the Transformer model provider during inference. With the increasing public attention on data privacy, the above data leakage problem has triggered researchers' study on secure Transformer inference. Implementing secure Transformer inference with secure multi-party computation (MPC) is today's hot topic. Due to the widely existence of non-linear functions in Transformer, it is hard to use MPC to implement secure Transformer inference, which leads to huge computation and communication cost. We focus on Softmax attention, bottleneck in secure Transformer inference, and propose two kinds of MPC-friendly attention mechanism, Softmax freeDiv Attention and 2Quad freeDiv Attention. By replacing the Softmax attention in Transformer with the MPC-friendly attention mechanism proposed, combining with the replacement of activation function GeLU and knowledge distillation, we propose an MPC-friendly Transformer convert framework, which can convert Transformer model to an MPC-friendly one, so as to improve the performance of secure Transformer inference later. Based on the proposed MPC-friendly Transformer convert framework , we perform secure Bert-Base inference on SST-2 in the LAN setting, using privacy computing protocols provided by secure processing unit (SPU). The result shows that the secure inference achieves 2.26 times speedup while maintaining the accuracy with non-approximation model.

Key words secure inference; Transformer; secure multi-party computation (MPC); secure processing unit (SPU); knowledge distillation

摘 要 Transformer 模型在自然语言处理、计算机视觉等众多领域得到了广泛应用,并且有着突出的表现.在 Transformer 的推理应用中用户的数据会被泄露给模型提供方.随着数据隐私问题愈发得到公众的关注,上述数据泄露问题引发了学者们对 Transformer 安全推理的研究,使用安全多方计算(secure multi-party computation, MPC)实现 Transformer 模型的安全推理是当前的一个研究热点.由于 Transformer 模型中存在大量非线性函数,因此使用 MPC 技术实现 Transformer 安全推理会造成巨大的计算和通信开销.针对

收稿日期:2023-12-01;**修回日期**:2024-03-11
基金项目:国家重点研发计划(2021YFB2700200);国家自然科学基金项目(U21B2021, 61972018, 61932014)
This work was supported by the National Key Research and Development Program of China (2021YFB2700200) and the National Natural Science Foundation of China (U21B2021, 61972018, 61932014).
通信作者:郭华(hguo@buaa.edu.cn)

Transformer 安全推理过程中开销较大的 Softmax 注意力机制，提出了 2 种 MPC 友好的注意力机制 Softmax freeDiv Attention 和 2Quad freeDiv Attention. 通过将 Transformer 模型中的 Softmax 注意力机制替换为新的 MPC 友好的注意力机制，同时结合激活函数 GeLU 的替换以及知识蒸馏技术，提出了一个 MPC 友好的 Transformer 转换框架，通过将 Transformer 模型转化为 MPC 友好的 Transformer 模型，提高 Transformer 安全推理的效率. 在局域网环境下使用安全处理器（secure processing unit, SPU）提供的隐私计算协议，基于所提出的 MPC 友好的 Transformer 转换框架，在 SST-2 上使用 Bert-Base 进行安全推理. 测试结果表明，在保持推理准确率与无近似模型一致的情况下，安全推理计算效率提高 2.26 倍.

关键词 安全推理；Transformer；安全多方计算；安全处理器；知识蒸馏

中图法分类号 TP309

近年来机器学习领域的 Transformer 模型飞速发展，其凭借巨大的参数规模和特殊的网络结构在众多领域有着出色的表现，包括自然语言处理[1-2]（natural language processing, NLP）、计算机视觉[3-4]（computer vision, CV）等. 在 Transformer 的应用中，通常有 3 类参与方，第 1 类是数据提供方，其提供在模型训练过程中使用到的训练样本；第 2 类是模型提供方，其使用训练样本进行模型训练从而得到一个训练好的模型，之后可以提供推理服务；第 3 类是模型使用方，其申请使用模型提供方所提供的推理服务，输入数据并得到推理结果. 以 OpenAI 推出的 ChatGPT 为例，其数据来源于互联网上几乎所有的公开文本数据，因此数据提供方可看作是互联网上的所有用户，再由 OpenAI 作为模型提供方在庞大的训练数据下进行模型训练，得到 ChatGPT 模型，最后将其公开以供各用户使用，此时使用 ChatGPT 的用户身份即为模型使用方，如图 1 所示.

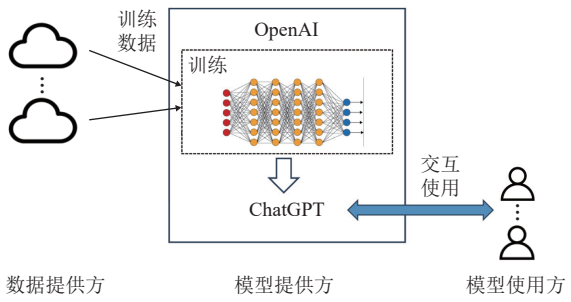

Fig. 1 Process of training and using ChatGPT

图 1 ChatGPT 训练及使用过程

近年来数据隐私安全问题逐渐得到社会关注，Transformer 模型的隐私保护问题成为一个研究热点[5-9]. 一方面，在模型训练过程中，模型提供方会收集大规模训练样本进行模型训练，但实际上这些训练样本有可能来自于互联网上的大量用户群体，或是某企业对应的用户群体，在这个过程中用户有权保护自己个人数据的隐私性. 另一方面，在模型推理过程中，用户更倾向于保护自己的输入数据及推理结果的隐私性，尤其是医疗诊断、图像识别等领域，用户的输入数据通常会涉及个人的隐私信息，同时模型提供方也需要保护自己的模型参数不会泄露，因为该模型是收集了大量的训练数据并消耗了大量的算力而训练得到的.

隐私保护机器学习（privacy preserving machine learning, PPML）使用同态加密（homomorphic encryption, HE）[10-11]、安全多方计算（secure multi- party computation, MPC）[9,12-13]、差分隐私（differential privacy, DP）[14]、可信执行环境（trusted execution environment, TEE）[15] 等技术对机器学习模型进行隐私保护，可以保证在训练过程中数据提供方的原始数据不会泄露给模型提供方，在推理过程中模型提供方的模型参数和用户的输入输出数据不会被泄露. 对于决策树[16]、支持向量机[17]、神经网络[12-13,18-19] 等机器学习算法的隐私保护技术已有大量研究.

近年来随着 Transformer 的提出，以 ChatGPT 为代表的 Transformer 模型愈发得到人们的关注. 在 Transformer 推理中，模型提供方希望模型参数不会泄露，模型使用方希望其模型推理的输入和输出不会泄露. 因此，Transformer 安全推理问题引发了众多学者的研究[6-9,20-22]. MPC 是 Transformer 安全推理中常用的密码技术，但其带来的额外开销往往是难以接受的，例如，输入长度为 512 的序列，一个 12 层的 BERT-Base 模型在 MPC 系统中进行安全推理需要 59.0 s，而在明文下只需要不到 1 s[21]. 因此，如何在保证准确率不受影响的前提下减少 Transformer 安全推理所需的时间成为了研究热点.

机器学习模型可划分为线性函数和非线性函数，对于复杂的非线性函数的处理方法一般是使用一个新的函数 f' 来替换原复杂的非线性函数. 对于 f' 的选

择通常有 2 种思路: 一种是近似方法, 即通过函数拟合的方法得到一个分段多项式函数, 将其作为非线性函数的替换, 代表工作有 ABY3[18], Delphi[23] 等, 该方法能够保持较高的准确率; 另一种是启发式方法, 使用具有与原函数相似性质的 MPC 友好函数进行替换, 例如使用 Quad 替换 GeLU, 代表工作有 MPCFormer[21], SecureML[12] 等, 这类方法相比于前一种方法在计算效率上通常会更加高效, 但需要额外考虑准确率下降的问题, 因此需要在后续添加微调步骤, 以保证模型的可用性.

目前对于 Transformer 安全推理工作的一大研究方向是将 Transformer 安全推理中开销较大的函数替换为 MPC 友好的函数, 再通过知识蒸馏等技术提升模型准确率, 最后在 MPC 友好的 Transformer 模型上实现安全推理. 以 Transformer 模型的一种变体 BERT-Base 为例, 在 BERT-Base 模型的基于 MPC 技术的安全推理中, Softmax 和 GeLU 占据了大量运行时间[21]. 例如文献 [21] 中指出对于一个 12 层的 BERT-Base 模型, 在输入序列长度为 512 时, Softmax 和 GeLU 耗时分别占总时长的 67.8% 和 18.6%, 二者占据了安全推理过程中 86.4% 的时长, 因此在 Transformer 安全推理中对 Softmax 注意力机制和激活函数 GeLU 的效率进行优化, 可以有效减少 Transformer 安全推理的总执行耗时. 目前对于 MPC 友好的函数替换方法的主要研究为如何在权衡效率和准确率的同时选择合适的函数替换方案.

文献 [21] 对 Transformer 模型进行了函数替换, 针对 Softmax 注意力机制首先选择 2ReLU Attention[12] 的替换方法并进行测试, 在其基础上又提出了 2Quad Attention 的替换方法; 针对激活函数 GeLU, 文献 [21] 提出了平方函数 Quad 的替换方法. 文献 [22] 对 Vision Transformer 模型进行了函数替换, 针对 Softmax 注意力机制选择并测试了目前已有的一系列注意力机制替换方法, 包括: 2ReLU Attention[12], 2Quad Attention[21], Scaling Attention[24], Linformer Attention[25] 等, 在其中选择了 2ReLU Attention, Scaling Attention 这 2 种作为其替换函数.

然而, 文献 [21−22] 中提出的函数替换方法并没有达到效率和准确率的平衡. Scaling Attention 等替换方法效率较高, 但准确率下降较大; 2ReLU Attention 和 2Quad Attention 准确率较高, 但同时也导致了较大的开销. 本文针对 Softmax 注意力机制给出一种新的函数替换方法, 相比 2ReLU Attention, 2Quad Attention 可以省去非 MPC 友好的倒数运算, 同时能保持较高的准确率, 达到了效率与准确率的平衡.

本文的主要贡献包括 2 个方面:

1) 提出了 2 种新的 MPC 友好的注意力机制 Softmax freeDiv Attention 和 2Quad freeDiv Attention, 在几乎不影响准确率的前提下减少了 Transformer 中的注意力机制在 MPC 场景中的计算量, 进而提高了基于 MPC 技术的 Transformer 安全推理速度;

2) 将提出的 MPC 友好的注意力机制与知识蒸馏技术结合, 提出了一个 MPC 友好的 Transformer 转换框架, 用于将 Transformer 模型转换为 MPC 友好的 Transformer 模型. 对转换得到的 Transformer 模型在隐语安全处理器(secure processing unit, SPU)下进行安全推理, 在 SST-2 任务上可以在保证准确率相比于无近似模型不受影响的前提下将安全推理速度提升 2.26 倍.

1 相关工作

1.1 Transformer 安全推理

目前 Transformer 安全推理工作的主要研究目标是在保证准确率的前提下提高计算效率. 在函数近似方法方面, 2022 年 Hao 等人[6] 提出的 Iron 基于同态加密和秘密分享技术实现各函数的隐私保护协议, 从而构建了 Transformer 安全推理框架, 对于其中的矩阵乘法, 在 Cheetah[13] 中基于同态加密方法的基础上进行优化, 达到了目前最优的开销. Iron 将非线性函数拆解为若干基础原语, 并调用 Sirnn[26] 实现. 2023 年 Zheng 等人[8] 提出的 Primer 直接将 Transformer 的非线性算子使用混淆电路实现. 同年 Puma[27] 和 Privformer[28] 分别基于 ABY3[18] 和 Falcon[29] 框架实现了 Transformer 的安全推理, 其主要工作是针对 Transformer 中独有的算子利用对应框架提供的协议设计对应的隐私保护协议. 2023 年文献 [20] 对 GeLU 进行分段拟合, 使用 Sirnn 中的多项技术来优化近似多项式的安全计算. 同年, 文献 [9] 基于函数秘密分享技术构造了 GeLU, Softmax 等非线性函数的安全计算协议, 大大提升了安全推理的效率.

在启发式方法方面, 2022 年 Chen 等人[7] 提出的 THE-X 对于非线性部分使用 ReLU 函数替换 GeLU, 使用一个 3 层全连接网络近似 Softmax, 但其中 ReLU 的计算则由数据持有者在明文上计算, 从而导致中间结果泄露给数据持有者, 进而可能导致模型持有者的模型参数相关信息遭到泄露. 同时 THE-X 使用知识蒸馏技术对模型进行了微调, 提升了模型准确率.

同年，Li 等人[21]提出 MPCFormer，该模型的主要思想是首先将 Transformer 中开销较大的函数即 Softmax 和 GeLU 替换为 MPC 友好的函数，之后使用知识蒸馏技术对该模型进行微调，提升模型准确率，最后使用现有 MPC 技术实现更新后的模型的安全推理. MPCViT[22]的思路与 MPCFormer 略有不同，其指出在 Vision Transformer（ViT）中并不是所有的注意力机制都同样重要，于是首先分析了多种注意力机制的效率与准确率（效率较高的注意力机制通常具有较低的准确率），将所有的 Softmax 注意力机制替换为一种准确率相对较高但效率相对较低的 MPC 友好的注意力机制 2ReLU Attention；之后使用神经架构搜索（neural architecture search, NAS）技术将部分 2ReLU Attention 注意力机制替换为准确率相对较低但效率相对较高的注意力机制 Scaling Attention，保证在不过多降低准确率的前提下尽可能达到最高的计算效率，最后再使用知识蒸馏技术提升模型准确率.

1.2 MPC 友好的函数替换方法

在现有的基于 MPC 的 PPML 中，有许多研究通过 MPC 友好的函数替换来加快安全推理速度. 早在 2017 年就有 SecureML 方案[12]将 Softmax 中的指数函数替换为 ReLU，因为在其针对的机器学习模型中 Softmax 仅在最后一层被使用，因此其对准确率几乎不造成影响. 2018 年 Chou 等人[30]提出的 Faster Cryptonets 使用二次函数替换了神经网络中的 ReLU，但其会导致梯度下降偏离，从而导致模型准确率大幅度下降. 2020 年的 Delphi 方案[23]通过 NAS 技术将神经网络中的部分 ReLU 替换为平方函数.

在 Transformer 安全推理的工作中，THE-X 方案[7]对于 Softmax 使用一个 3 层全连接网络来进行替换. MPCFormer[21]将 Transformer 中的 Softmax 注意力机制和激活函数 GeLU 分别替换为 MPC 友好的注意力机制和激活函数. 通过实验，MPCFormer 得出在 GLUE 数据集下分别将注意力机制和激活函数 GeLU 替换为 MPC 友好的 2Quad Attention 注意力机制和平方函数 Quad，在准确率下降幅度不大的前提下获得 2.2 倍的安全推理速度. MPCViT[22]针对 ViT，指出 THE-X[7]和 MPCFormer[21]中的注意力机制替换方法并不灵活，直接应用在 ViT 中会导致准确率大幅度下降. 因此，针对不同特点的注意力机制替换方法，MPCViT 使用 NAS 技术选择对 Transformer 中注意力机制的不同头级（head-wise）使用不同的替换，而不是像 MPCFormer 中对所有的注意力机制使用相同的替换.

2 预备知识

在本节中，首先介绍 Transformer 和知识蒸馏的概念，然后介绍 Transformer 安全推理框架.

2.1 Transformer

Transformer 采用 Encoder-Decoder 结构，由于本文主要针对 BERT 这类仅包含 Encoder 的 Transformer 模型（Encoder-only Transformer）进行研究，本节以 BERT 模型为例，介绍 Encoder-only Transformer 结构，如图 2 所示.

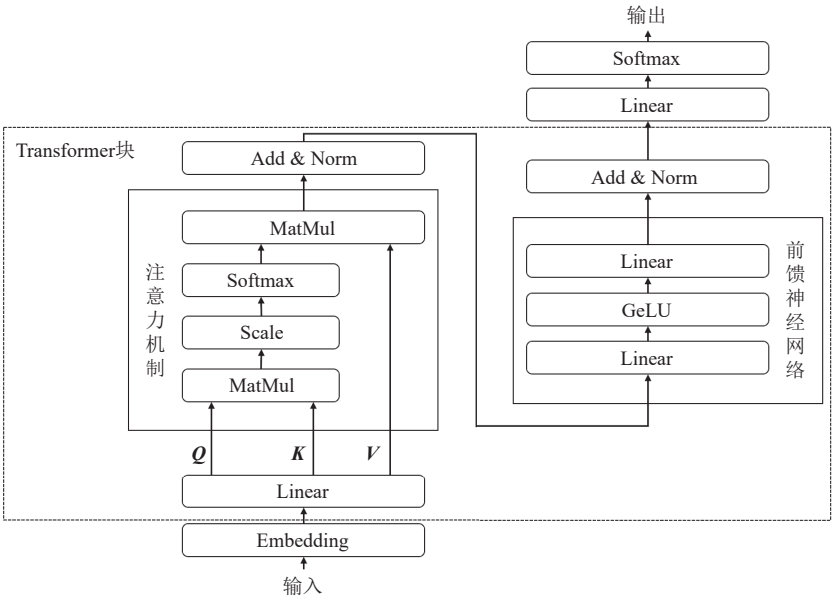

Fig. 2 Architecture of Encoder-only Transformer

图 2 Encoder-only Transformer 结构

Transformer 的输入首先会经过一个 Embedding 层, 输出词向量 $\boldsymbol{X}$, 之后的 Encoder 部分包含多个 Transformer 块, 在每个 Transformer 块中包含 3 部分: 注意力机制(*Attention*)、前馈神经网络(feed forward network, *FFN*)、2 个层归一化(*LN*). 这 3 部分详细介绍如下:

1)注意力机制(*Attention*).注意力机制有助于模型关注输入序列中的不同部分, 输入矩阵 $\boldsymbol{Q}=\boldsymbol{X}\boldsymbol{W}_{\boldsymbol{Q}}$, $\boldsymbol{K}=\boldsymbol{X}\boldsymbol{W}_{\boldsymbol{K}}$, $\boldsymbol{V}=\boldsymbol{X}\boldsymbol{W}_{\boldsymbol{V}}$, 其中 $\boldsymbol{W}_{\boldsymbol{Q}}$, $\boldsymbol{W}_{\boldsymbol{K}}$, $\boldsymbol{W}_{\boldsymbol{V}}$ 表示权值矩阵, 令 d 表示向量维度, 则注意力机制如式(1)所示.

$$Attention(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V})=softmax\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d}}\right)\boldsymbol{V}, \tag{1}$$

其中 *softmax* 如式(2)所示.

$$softmax(x_0,x_1,\cdots,x_{n-1})=\frac{1}{\sum_{i=0}^{n-1}\mathrm{e}^{x_i}}(\mathrm{e}^{x_0},\mathrm{e}^{x_1},\cdots,\mathrm{e}^{x_{n-1}}). \tag{2}$$

多头注意力机制(multi-head attention, MHA)将上述注意力机制扩展到 H 个并行的注意力层, 如式 (3)所示.

$$MultiHead(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V})=Concat(Attention(\boldsymbol{Q}_i,\boldsymbol{K}_i,\boldsymbol{V}_i))\boldsymbol{W}^{O},i=1,2,\cdots,H. \tag{3}$$

2)前馈神经网络(*FFN*). *FFN* 包含 2 个线性层和中间的一个激活函数, 如式(4)所示.

$$FFN(\boldsymbol{X})=GeLU(\boldsymbol{X}\boldsymbol{W}_1+\boldsymbol{B}_1)\boldsymbol{W}_2+\boldsymbol{B}_2, \tag{4}$$

其中 GeLU 近似如式(5)所示.

$$GeLU(x)=0.5x\left[1+\tanh\left(\sqrt{\frac{2}{\pi}}\left(x+0.047715x^3\right)\right)\right]. \tag{5}$$

3)层归一化(*LN*). *LN* 需计算输入的均值和方差, 对于输入 $\boldsymbol{x}=(x_0, x_1, \cdots, x_{n-1})$, LN 如式(6)所示.

$$y_j=\gamma_j\cdot\frac{x_j-\mu}{\sqrt{\sigma^2+\varepsilon}}+\beta_j,\quad \mu=\frac{1}{n}\sum_{j=0}^{n-1}x_j,\ \sigma^2=\frac{1}{n}\sum_{j=0}^{n-1}(x_j-\mu)^2. \tag{6}$$

2.2 知识蒸馏

知识蒸馏[31]是模型压缩技术中的一种, 用于解决小的模型训练困难的问题. 知识蒸馏技术允许一个小的学生模型 S 学习一个大的教师模型 T 的知识. 具体来说, 学生模型在训练过程中会去模仿教师模型的行为. 教师模型的行为能提供比训练数据标签更多的信息, 学生模型通过学习这些信息能够获得教师模型的泛化能力. 模型的行为可以用行为函数描述, 在 Transformer 模型的蒸馏中, MHA 层、*FFN* 层或者一些中间表示层都可以看作行为函数. 用 $f^{\mathrm{S}}, f^{\mathrm{T}}$ 分别表示学生模型和教师模型的行为函数, 则知识蒸馏的过程可以描述为最小化目标函数式 (7).

$$L_{\mathrm{KD}}=\sum_{\boldsymbol{d}_i\in D}L(f^{\mathrm{S}}(\boldsymbol{d}_i),f^{\mathrm{T}}(\boldsymbol{d}_i)), \tag{7}$$

其中 $L(\cdot)$ 是评估教师模型与学生模型之间差异的损失函数, D 表示训练数据集, $\boldsymbol{d}_i$ 是一条文本输入.

2.3 Transformer 安全推理框架

本文沿用文献 [21] 所给出的 Transformer 安全推理框架, 整体架构如图 3 所示, 主要分为 3 个部分: 函数替换、知识蒸馏、安全推理, 其中前 2 部分共同组成了 MPC 友好的 Transformer 转换框架.

1)函数替换. 输入预训练的 Transformer 模型, 将其中开销较大的函数替换为 MPC 友好的函数, 该步骤可提升后续安全推理的推理速度.

2)知识蒸馏. 以无近似模型为教师模型, 替换后的模型为学生模型, 使用知识蒸馏技术对替换后的模型进行微调, 得到更新参数后的 MPC 友好的近似模型, 该步骤可以提升后续安全推理的准确率.

3)安全推理. 服务端以更新参数后的 MPC 友好的近似模型作为输入, 客户端以其隐私数据作为输入, 双方在不泄露各自输入的安全保证下执行 MPC 协议进行安全推理.

3 MPC 友好的 Transformer 转换框架

原始 Transformer 预训练模型在准确率上表现良好, 但由于其中的 Softmax 注意力机制和激活函数 GeLU 在使用 MPC 技术实现时效率较低, 导致模型推理速度较慢. 本节提出一种 MPC 友好的 Transformer 转换框架, 该框架可将预训练的 Transformer 模型转换为 MPC 友好的 Transformer 模型, 同时保证模型准确率不受影响, 并且提高安全推理的推理速度, 从而可用于 Transformer 安全推理.

3.1 概 述

MPC 友好的 Transformer 转换框架分为函数替换和知识蒸馏 2 个阶段, 工作流程如图 4 所示. 第 1 阶段为函数替换, 将 Softmax 注意力机制替换为本文提出的 MPC 友好的 Softmax freeDiv Attention 或 2Quad freeDiv Attention(两者统称为 freeDiv Attention), 同时将前馈神经网络中的激活函数 GeLU 替换为 MPC 友好的 ReLU 函数或平方函数 Quad, 可提高推理速度. 由于该替换改变了注意力机制和激活函数的功能, 直接使用原始模型的参数会导致准确率下降. 第 2 阶

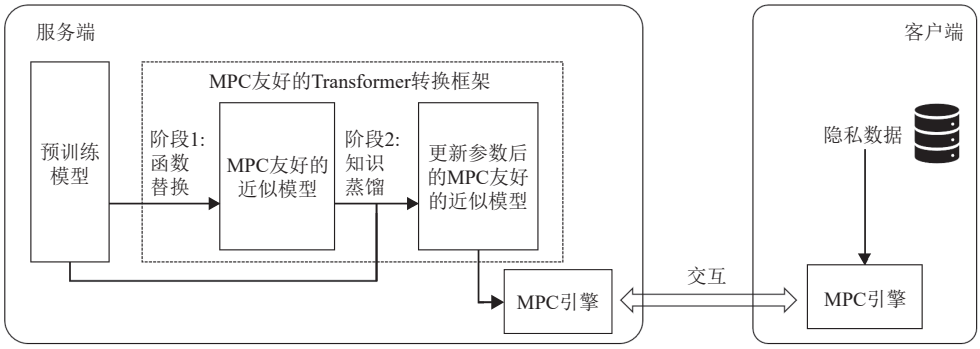

Fig. 3 Secure Transformer inference framework

图 3 Transformer 安全推理框架

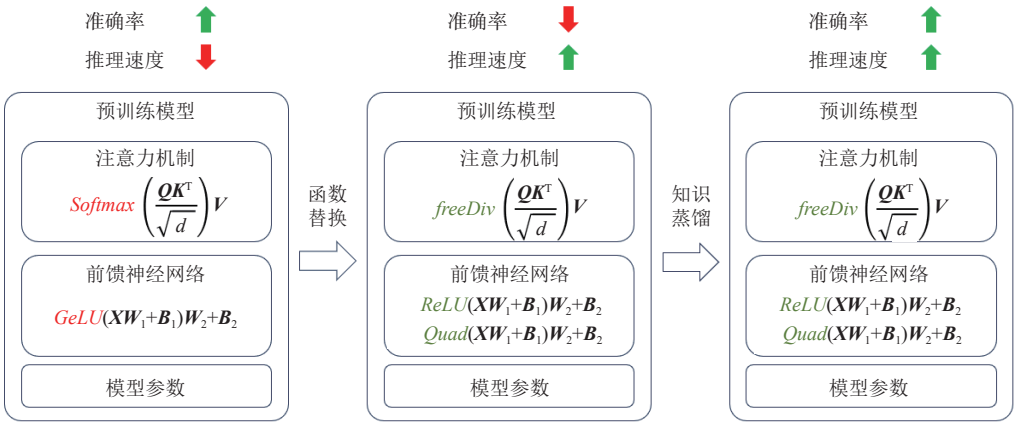

Fig. 4 MPC-friendly Transformer convert framework

图 4 MPC 友好的 Transformer 转换框架

段的知识蒸馏技术以原模型为教师模型，以上一步替换后的模型为学生模型，对替换后的模型使用知识蒸馏技术进行微调，更新模型参数使其与新的注意力机制和激活函数适配，从而提高了模型的准确率.

通过上述 2 个阶段，本文提出的 MPC 友好的 Transformer 转换框架实现了预训练 Transformer 模型到 MPC 友好的 Transformer 模型的转换.

3.2 函数替换

在函数替换阶段，首先分析在 Transformer 安全推理过程中的主要开销，通过将开销较大的函数替换为 MPC 友好的函数可以提高安全推理的效率. MPCFormer[21] 和 MPCViT[22] 分别通过实验指出 Softmax 和 GeLU 占据了 Transformer 推理的大部分时长. 因此本文主要考虑分别将 Softmax 注意力机制和激活函数 GeLU 替换为 MPC 友好的注意力机制和激活函数. 对于 Softmax 注意力机制，其中包含 3 种非线性运算：指数运算、比较运算和倒数运算，均需通过 MPC 协议安全计算. 文献 [21] 中给出的 2Quad Attention 替换方案去除了其中的指数运算和比较运算，并且在 SST-2 任务上达到较好的准确率，但其中还存在较大开销的倒数运算. 本文考虑使用一个可以在明文下计算的函数得到倒数运算中的输入，从而去除安全倒数运算，进一步减少开销. 对于激活函数 GeLU，文献 [21] 中已经给出性能和准确率都表现较优的替换函数，本文将直接使用这些函数进行替换. 具体的替换方案将在第 4 节介绍.

3.3 知识蒸馏

在知识蒸馏阶段中，本文使用无近似模型 M 作为教师模型，近似后的模型 M' 作为学生模型，在下游任务数据集上进行知识蒸馏. 初始化时学生模型 M' 使用与教师模型 M 相同的模型参数，二者仅在使用的函数上不同.

蒸馏任务执行时分为 2 个步骤：在第 1 步中学生模型主要学习教师模型在 Transformer 块的行为；在第 2 步中学生模型主要学习教师模型在最后的预测层的行为. 下面具体描述 2 个步骤：

第 1 步，学生模型将学习教师模型在每个 Transformer 块的注意力矩阵以及每一个 Transformer 块之

后的隐藏状态信息，这些信息中包含了大量语言知识. 蒸馏过程中向教师模型和学生模型输入同样一组训练数据，并分别计算 2 个模型在每一层的注意力矩阵以及每一层之后的隐藏状态信息. 使用均方误差作为损失函数，计算如式(8)所示.

$$L_1=\sum_{\boldsymbol{d}_i\in D}(MSE(\boldsymbol{att}^{\mathrm{S}},\boldsymbol{att}^{\mathrm{T}})+MSE(\boldsymbol{rep}^{\mathrm{S}},\boldsymbol{rep}^{\mathrm{T}})),\tag{8}$$

$$MSE(\boldsymbol{Y},\hat{\boldsymbol{Y}})=\frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{Y}_i-\hat{\boldsymbol{Y}}_i)^2,$$

其中$\boldsymbol{att}^{\mathrm{S}}$，$\boldsymbol{att}^{\mathrm{T}}$分别表示学生模型和教师模型的注意力矩阵，$\boldsymbol{rep}^{\mathrm{S}}$，$\boldsymbol{rep}^{\mathrm{T}}$分别表示学生模型和教师模型在 Transformer 块之后的隐藏状态信息.

第 2 步，学生模型将学习教师模型在最后的预测层输出. 同样在蒸馏过程中向教师模型和学生模型输入同样一组训练数据并分别计算这 2 个模型在预测层的输出. 损失函数设置为学生模型 logits 输出和教师模型 logits 输出之间的软交叉熵损失，如式(9)所示.

$$L_2=CE(\boldsymbol{logit}^{\mathrm{T}}/t,\boldsymbol{logit}^{\mathrm{S}}/t),\tag{9}$$

其中$\boldsymbol{logit}^{\mathrm{T}}$和$\boldsymbol{logit}^{\mathrm{S}}$分别表示学生模型和教师模型的 logits 向量，CE 表示交叉熵损失，t 是温度. 在本文的实验过程中，设置 $t=1$ 时有较好的蒸馏效果.

4 MPC 友好的函数替换方案

本文针对 Transformer 模型中的 Softmax 注意力机制和激活函数 GeLU 给出 MPC 友好的函数替换方案. 对 Softmax 注意力机制替换时，本文提出了 2 种新的注意力机制 Softmax freeDiv Attention 和 2Quad freeDiv Attention，其中 2Quad freeDiv Attention 在计算效率上可持平目前计算效率最高的注意力机制 Scaling Attention，并且不会导致 Scaling Attention 中存在的准确率大幅度下降问题. 对激活函数 GeLU 替换时，本文使用文献 [21] 中提到的 2 种替换方案.

4.1 注意力机制的替换

对于注意力机制的替换，以往采取的方法包括：将 Softmax 表达式中的指数函数替换为 ReLU 的 2ReLU Attention、将指数函数替换为平方函数的 2Quad Attention、将整个注意力机制替换为矩阵乘法的 Scaling Attention 等，其中$ReLU(\hat{x})=\max(\hat{x},0)$.

上述替换在准确率和效率上各有优劣，如 2ReLU Attention 相比于原始的 Softmax Attention 不会导致准确率大幅度下降，但会使计算效率降低. Scaling Attention 的表达式为

$$ScaleAttn(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V})=\frac{1}{n}\left(\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}\right)\boldsymbol{V},\tag{10}$$

其中 n 为输入序列长度. 与式(1)相比，式(10)中省去 Softmax 的计算过程，因此计算高效，但会导致推理阶段准确率大幅度降低.

MPCViT[22] 中总结了 Softmax 注意力机制的 3 个重要特性，即单调性、非负性、求和为 1. 上述 2ReLU Attention、2Quad Attention 的主要思想是将注意力机制中的指数函数替换为计算较为简单的其他函数. 因为在基于 MPC 技术的 Transformer 安全推理中通常要将计算指数函数近似为分段多项式，再使用不经意分段多项式求值，需调用多次安全比较协议和安全乘法协议，而将指数函数替换为 ReLU 或平方函数分别只需要 1 次比较+1 次乘法和 1 次乘法，大大提高了安全推理的效率. 其中的 ReLU 和平方函数本身均能保证单调性和非负性(其中单调性只需要在数据分布区间上满足)，最后再通过将每一项除以所有项求和的方式来满足求和为 1.

目前的注意力机制替换方法将指数函数替换为 MPC 友好的 ReLU 或平方函数，能够在满足 Softmax 注意力机制的前 2 个性质的前提下大大减少运算量，从而提高计算效率. 但为了满足第 3 个性质，需将每一项除以所有项的求和. 对于一个规模为 n 的张量，使用 MPC 技术时需执行 n 次秘密值之间的除法. 由于在 MPC 中除法的计算开销远大于加法，因此通常先计算所有项求和的倒数，再计算 n 次乘法，共需 1 次除法与 n 次乘法的开销. 若使用一个常数来近似分母(即各分项的求和)，将注意力机制的第 3 个特性“求和为 1”放宽到“求和近似于 1”，那么可以在保证准确率的前提下去除倒数运算，减少开销，然而难点在于如何选择这个近似的常数.

在选择注意力机制中分母近似常数时，本文首先使用预训练的 Transformer 模型对数据集进行多次推理，将每次推理过程中不同输入序列长度下的分母均值进行记录，利用这些数据可以拟合得到一个关于输入序列长度的函数 f. 在给定输入 $\boldsymbol{x}$ 的情况下，本文提出的方案将使用 $f(|\boldsymbol{x}|)$ 作为分母近似常数，其中 $|\boldsymbol{x}|$ 是输入序列长度.

具体来说，本文使用预训练的 Transformer 模型对 GLUE 数据集中的若干条输入进行推理，给出测试数据集中不同的输入序列长度和 Softmax 注意力机制中分母均值的关系，如图 5 所示，横坐标的输入序列长度等于 Softmax 函数输入张量的维度，即为 Softmax 函数中分母求和时的项数，纵坐标表示对应

的 Softmax 注意力机制中的分母均值.

图 5 表明分母均值并不随着输入序列长度的增加而线性增长. 本文使用幂函数 $f_{\text{softmax}}(\hat{x}) = a\hat{x}^b$ 对其进行拟合, 其中 a=0.992 393 766 677 209 6, b=0.333 253 864 762 922. 得到的拟合结果如图 6 所示.

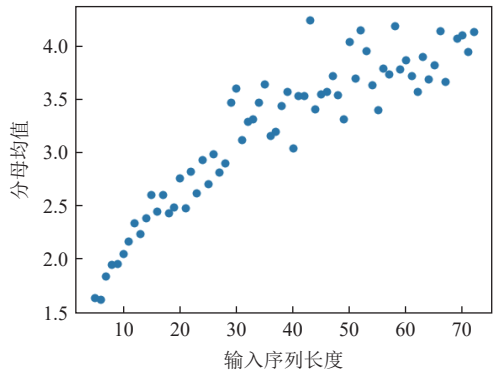

Fig. 5 Scattergram of the mean of denominator in Softmax attention mechanism

图 5 Softmax 注意力机制中分母均值的散点图

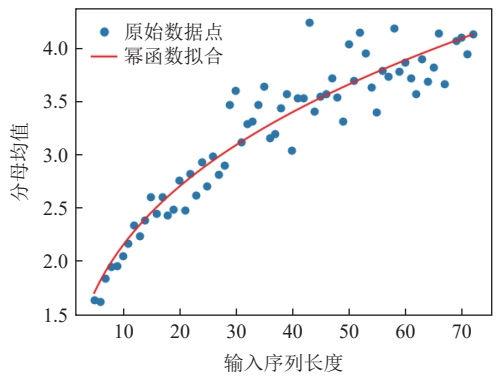

Fig. 6 Function fitting of the mean of denominator in Softmax attention mechanism

图 6 Softmax 注意力机制中分母均值的函数拟合

得到的 Softmax freeDiv Attention(简写为 *Soft_freeDivAttn*) 表达式如式(11)所示.

$$Soft\_freeDivAttn(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = Softmax_{\text{freeDiv}}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d}}\right)\boldsymbol{V},$$
$$Softmax_{\text{freeDiv}}(\boldsymbol{x}) = \frac{(\mathrm{e}^{x_0-\max},\cdots,\mathrm{e}^{x_{n-1}-\max})}{f_{\text{softmax}}(|\boldsymbol{x}|)}, \tag{11}$$

其中, f_{softmax} 表示在 Softmax Attention 中从 $|\boldsymbol{x}|$ 到分母的近似常数的映射函数, 其中 $|\boldsymbol{x}|$ 在进行 Transformer 推理时可获得, f_{softmax} 不依赖推理阶段的具体输入, 可视为模型参数的一部分.

上述 Softmax freeDiv Attention 的思想及方法可以与其他注意力机制结合以获得更好的效果. 在 MPCFormer 中指出 2Quad Attention 在准确率和效率上都表现较为良好, 因此可同样将 Softmax freeDiv Attention 的思想与 2Quad Attention 结合得到 2Quad freeDiv Attention, 同时需对 Softmax freeDiv Attention 做 2 项修改:

1)根据预训练 Transformer 模型在 2Quad Attention 注意力机制下测试不同输入序列长度时 2Quad Attention 注意力机制的分母数值, 拟合得到二者之间的映射函数 f_{2Quad};

2)将 f_{2Quad} 与 2Quad Attention 结合, 得到 2Quad freeDiv Attention(简写为 *2Quad_freeDivAttn*) 的表达式如式(12)所示.

$$2Quad\_freeAttn(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = 2Quad_{\text{freeDiv}}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d}}\right)\boldsymbol{V},$$
$$2Quad_{\text{freeDiv}}(\boldsymbol{x}) = \frac{((x_0+c)^2,\cdots,(x_{n-1}+c)^2)}{f_{\text{2Quad}}(|\boldsymbol{x}|)}. \tag{12}$$

对于 2Quad freeDiv Attention, 按照上述 1)中的方法得到的 f_{2Quad} 为图 7 中拟合的幂函数, 其中 a= 84.073 737 580 711 03, b=0.717 474 525 577 988 7, c=5.

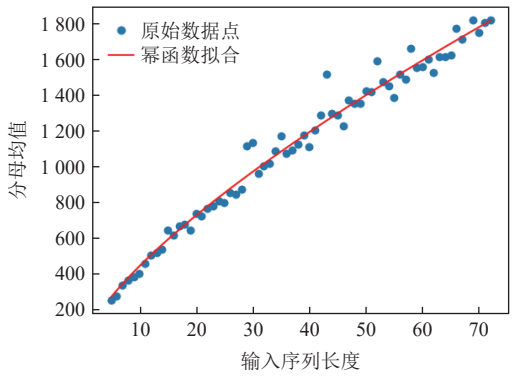

Fig. 7 Function fitting of the mean of denominator in 2Quad attention mechanism

图 7 2Quad 注意力机制中分母均值的函数拟合

对于本文提出的 2 种注意力替换机制 Softmax freeDiv Attention 和 2Quad freeDiv Attention, 分别相比于 Softmax Attention 和 2Quad Attention 减少了 1 次除法和 n 次乘法的开销. 表 1 中给出各种注意力机制在 ABY3 框架下的计算延迟和速度提升倍数, 本文提出的 2 种替换方案在延迟上均表现较好.

第 5 节中将会对这 2 种注意力机制以及以往工作中使用的注意力机制在效率和准确率上的表现进行更进一步的测试与比较.

4.2 激活函数 GeLU 的替换

对于激活函数 GeLU 的替换, 以往方案中采取的较为高效的 2 种替换方式包括: 将 GeLU 替换为 ReLU、将 GeLU 替换为平方函数. 此外有的方案还会对 GeLU 用分段多项式进行精确近似, 在准确率几乎不受影

Table 1 Secure Computational Latency of Different Attention Mechanisms

表 1 不同的注意力机制的安全计算延迟

注意力机制	延迟/s	速度提升倍数
Softmax Attention	0.236	1
2ReLU Attention	0.159	1.48
2Quad Attention	0.157	1.50
Scaling Attention	0.060	3.93
Softmax freeDiv Attention（本文）	0.153	1.54
2Quad freeDiv Attention（本文）	0.064	3.69

响的同时导致较低的计算效率. 由于在本文框架中后续会使用知识蒸馏技术提高模型的准确率, 因此在权衡准确率和计算效率时可适当放宽对于准确率的要求从而追求更高的计算效率, 不考虑对 GeLU 使用分段多项式进行精确近似的方案, 而是选择 ReLU 和平方函数 Quad 这 2 种替换方法.

对于 ReLU 和 Quad 这 2 种激活函数对 GeLU 的替换, 虽然 Quad 相比于 ReLU 少了 1 次比较运算, 在计算上稍加高效, 但由于同时进行了一些其他注意力机制的替换, 使用 Quad 对 GeLU 进行替换相比于 ReLU 会导致准确率表现稍差, 第 5 节中将对二者进行具体实验与说明.

通过上述分析, 在注意力机制替换方面, 本文选择使用 Softmax freeDiv Attention 或者 2Quad freeDiv Attention 这 2 种注意力机制, 在对激活函数 GeLU 的替换方面, 选择使用 ReLU 或者 Quad 来进行替换.

5 实验与结果

本节对 MPC 友好的 Transformer 转换框架在 Transformer 安全推理中应用时的效果进行实验与分析.

5.1 实验设置

1)实验环境和 MPC 框架配置

实验统一在局域网下进行测试, 操作系统为 Ubuntu 20.04, 处理器为 Interl® Core™ i9-10900X 3.70 GHz, 知识蒸馏步骤使用 GPU 为 NVIDIA Tesla P100 PCIe 16 GB.

在 Transformer 安全推理中, 本实验借助于开源隐语框架 SPU v0.6.0b0[32] 来实现, 这是一个通用的可用于隐私保护机器学习的框架. 在其中选择使用半诚实安全的 ABY3 协议[18] 来实现 Transformer 安全推理.

2)模型结构和数据集

本文实验使用模型结构 BERT-Base-Cased, 使用数据集 GLUE, 针对的任务包括 SST-2, STS-B, RTE. 在进行安全推理的延迟测试时, 使用的输入序列长度为 128. 由于 Softmax 函数的输入规模与输入序列长度的平方成正比, 因此对于 Softmax 注意力机制的替换的加速效果会随着输入序列长度的增加而愈发显著. 为了后续与 MPCFormer[21] 对比, 本文选择的输入序列长度与 MPCFormer 实验中相同, 均为 128.

3)知识蒸馏配置

本文实验在进行知识蒸馏时分为 2 步: 第 1 步是对 Transformer 块进行蒸馏, 使用学习率为 5E−5, 共迭代 10 次; 第 2 步是对预测层进行蒸馏, 使用学习率为 1E−5, 共迭代 5 次.

5.2 不同替换方案的效果测试

本节测试不同注意力机制替换方案与 GeLU 替换方案在准确率和效率上的表现, 并将其与原始未进行替换的 Transformer 模型的基准方案进行对比. 选择的模型基础为 BERT-Base-Cased, 使用的数据集为 GLUE, 针对的任务为 SST-2, STS-B, RTE, 所使用的输入序列长度选择为 128, 主要对比的是各替换方案相比于基准值的相对值.

在注意力机制的替换方面, 本文实验测试了以往方案中常用的多种注意力机制替换方法 2ReLU Attention, 2Quad Attention, Scaling Attention 及本文提出的 Softmax freeDiv Attention, 2Quad freeDiv Attention. 在激活函数 GeLU 的替换方面, 测试了将 GeLU 替换为 ReLU 和平方函数 Quad 两种方法. 同时将以上 2 种替换两两组合, 并与基准方案(即未替换的 Softmax 注意力机制和激活函数 GeLU)进行比较, 得到的结果如表 2 所示.

从表 2 可以看出, 在前 4 种注意力机制中, 速度提升方面表现最好的是 Scaling Attention, 其与激活函数的 Quad 替换方法组合达到了 2.26 倍的速度提升, 但其准确率大幅度下降. 而对于本文提出的 2 种注意力机制, 可以看出对于 2Quad freeDiv Attention + Quad 的组合在速度提升方面与目前效率提升最明显的 Scaling Attention 相当, 且在准确率方面表现良好, 相比于原模型没有准确率的下降, 因此在效率和准确率方面达到了目前所有方案中的最优.

表 3 给出了本文最终选择的表现最佳的 2Quad freeDiv Attention + Quad 的组合替换方案与原模型在准确率和安全推理速度上的对比. 可以看出, 使用本文提出的 MPC 友好的 Transformer 转换框架转换后的模型在安全推理速度上达到了 2.26 倍的提升, 且没有准确率损失, 达到了目前已知方案中的最优.

Table 2 Performance of Different Function Replacement Schemes Compared with the Benchmark

表 2 不同函数替换方案相比于基准方案的表现

注意力机制	激活函数	速度提升倍数	准确率/%
Softmax Attention	GeLU	1	91.5
	ReLU	1.36	92.0
	Quad	1.42	92.0
2ReLU Attention	GeLU	1.18	91.7
	ReLU	1.75	91.5
	Quad	1.78	90.9
2Quad Attention	GeLU	1.2	91.6
	ReLU	1.79	91.5
	Quad	1.84	91.8
Scaling Attention	GeLU	1.38	50.9
	ReLU	2.17	50.9
	Quad	2.26	50.9
Softmax freeDiv Attention（本文方法）	GeLU	1.17	91.7
	ReLU	1.59	91.7
	Quad	1.63	92.1
2Quad freeDiv Attention（本文方法）	GeLU	1.37	91.6
	ReLU	2.16	91.7
	Quad	2.26	91.8

Table 3 Performance of the Converted Model Compared with the Original Model

表 3 转换后的模型与原模型的性能对比

模型	速度提升倍数	准确率/%
原模型	1	91.5
转换后的 MPC 友好的模型	2.26	91.8

5.3 不同任务的效果测试

本节测试不同注意力机制替换方案与 GeLU 替换方案在不同任务下的准确率和效率表现，以表明本文提出的 MPC 友好的 Transformer 转换框架的泛化性. 选择的模型基础为 BERT-Base-Cased，使用的数据集为 GLUE，针对 GLUE 数据集中的 3 类任务各选择一个具体任务：单句任务 SST-2、相似性任务 STS-B、释义任务 RTE，所使用的输入序列长度选择为 128.

在注意力机制的替换方面，本实验选择了 Softmax Attention，2Quad Attention，Softmax freeDiv Attention，2Quad freeDiv Attention 这 4 种替换方法. 在激活函数 GeLU 的替换方面，本文实验选择了 Quad 的替换方法. 得到的结果如表 4 所示，其中 SST-2 和 RTE 任务评估指标为准确率，STS-B 任务评估指标为 Pearson 相关系数和 Spearman 相关系数的均值（后文简称为相关系数）.

Table 4 Performance of Different Function Replacement Schemes in Different Tasks

表 4 不同函数替换方案在不同任务中的表现

替换方案	任务	蒸馏前的评估指标/%	蒸馏后的评估指标/%
Softmax Attention +GeLU	SST-2	91.5	
	STS-B	88.7	
	RTE	63.2	
Softmax freeDiv Attention +GeLU（本文方法）	SST-2	91.3	91.6
	STS-B	84.0	89.0
	RTE	55.6	61.4
Softmax Attention +Quad	SST-2	50.9	92.0
	STS-B	4.2	89.2
	RTE	52.7	65.3
Softmax freeDiv Attention +Quad（本文方法）	SST-2	50.9	91.5
	STS-B	4.2	88.0
	RTE	52.7	47.3
2Quad Attention +Quad	SST-2	50.9	91.8
	STS-B	−5.7	84.5
	RTE	52.7	59.2
2Quad freeDiv Attention +Quad（本文方法）	SST-2	49.2	91.5
	STS-B	2.9	81.0
	RTE	52.0	54.9

从表 4 可以看出，与基准值（Softmax Attention + GeLU）相比，本文提出的 freeDiv Attention 应用于不同任务上均有较好的准确率表现. 对于单句任务 SST-2，freeDiv Attention 至多达到了 91.6%（上升 0.1 个百分点）的准确率；对于相似性任务 STS-B，freeDiv Attention 的相关系数至多达到了 89.0%（上升 0.3 个百分点）；对于释义任务 RTE，freeDiv Attention 至多达到了 61.4%（下降 1.8 个百分点）的准确率，因此本文提出的 freeDiv Attention 在不同任务下具有一定的泛化性.

5.4 消融实验

知识蒸馏过程是 MPC 友好的 Transformer 转换框架的重要组成部分，本文实验测试了知识蒸馏过程对于转换后模型的准确率影响，测试结果如表 4 所示.

由表 4 表明，对于 SST-2 任务，知识蒸馏过程至多可以提高 42.4 个百分点的准确率；对于 STS-B 任务，知识蒸馏过程至多可以提高 90.2 个百分点的相关系数；对于 RTE 任务，知识蒸馏过程至多可以提高 12.6 个百分点的准确率. 因此，知识蒸馏过程可以

有效提高转换后模型的准确率.

本文提出的 MPC 友好的 Transformer 转换框架对不同任务的 Transformer 模型进行转换所需时间,即知识蒸馏步骤所需时间,结果如表 5 所示.

Table 5 Time Consuming of MPC-friendly Transformer Convert Framework for Different Tasks

表 5 MPC 友好的 Transformer 转换框架针对不同任务消耗的时间

任务 (数据集大小)	耗时/min
SST-2 (67 000)	338
STS-B (57 000)	200
RTE (25 000)	46

5.5 与 MPCFormer 的对比

本节给出与本文使用同一安全推理框架(函数替换—知识蒸馏—安全推理)的 MPCFormer 方案[21]的对比结果,主要对比本文所提出的 MPC 友好的函数替换方案与 MPCFormer 中给出的表现最好的函数替换方案在同一数据集、同一任务上的准确率和延迟的表现. 对比结果如表 6 所示,基于本文的函数替换方案实现的 Transformer 安全推理在速度提升上优于 MPCFormer.

Table 6 Performance Comparison Between Our Work and Reference [21]

表 6 本文工作与文献 [21] 的性能对比

方案	准确率/%	速度提升倍数
无近似模型 Softmax Attention+GeLU	91.5 (+0.0)	1
2Quad Attention+Quad[21]	91.8 (+0.3)	1.78
2Quad freeDiv Attention+Quad (本文工作)	91.8 (+0.3)	2.26

注:括号内数字表示提升的百分点.

6 结 论

本文提出了 2 种可用于 Transformer 的注意力机制 freeDiv Attention 注意力机制,并将其与激活函数 GeLU 的替换、知识蒸馏技术结合提出了一个 MPC 友好的 Transformer 转换框架,可以将预训练的 Transformer 模型转化为 MPC 友好的 Transformer 模型用于安全推理,在保证准确率不受影响的同时提高安全推理效率. 利用本文的 MPC 友好的 Transformer 转换框架将一个预训练的 BERT-Base-Cased 模型转化为 MPC 友好的模型之后,在 GLUE 数据集上针对 SST-2 任务可在保持模型准确率不降低的前提下,提高推理速度 2.26 倍. 在本文提出的 2 种 freeDiv Attention 注意力机制中,拟合分母均值时使用的是整个模型中所有层的均值,但观察到对于 Transformer 的不同层,甚至是多头注意力机制中的不同头,其注意力机制中的分母均值有不同分布,下一步可考虑对其进行细化划分,在一些较为复杂的任务上可能会有更好的效果.

作者贡献声明: 刘伟欣和管晔玮提出了方案设计并撰写论文;霍嘉荣和丁元朝完成实验并撰写论文;郭华和李博提出指导意见并修改论文.

参 考 文 献

[1] Hoffmann J, Borgeaud S, Mensch A, et al. An empirical analysis of compute-optimal large language model training[J]. Advances in Neural Information Processing Systems, 2022, 35: 30016−30030

[2] Chan S, Santoro A, Lampinen A, et al. Data distributional properties drive emergent in-context learning in transformers[J]. Advances in Neural Information Processing Systems, 2022, 35: 18878−18891

[3] Liu Ze, Lin Yutong, Cao Yue, et al. Swin transformer: Hierarchical vision transformer using shifted windows[C]//Proc of the IEEE/CVF Int Conf on Computer Vision. Piscataway, NJ: IEEE , 2021: 10012−10022

[4] Liu Ze, Hu Han, Lin Yutong, et al. Swin transformer v2: Scaling up capacity and resolution[C]//Proc of the IEEE/CVF Conf on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2022: 12009−12019

[5] Jawalkar N, Gupta K, Basu A, et al. Orca: FSS-based secure training with GPUs[J]. Cryptology ePrint Archive, 2023

[6] Hao Meng, Li Hongwei, Chen Hanxiao, et al. Iron: Private inference on transformers[J]. Advances in Neural Information Processing Systems, 2022, 35: 15718−15731

[7] Chen Tianyu, Bao Hangbo, Huang Shaohan, et al. THE-X: Privacy-preserving transformer inference with homomorphic encryption[C]//Findings of the Association for Computational Linguistics: ACL 2022. Stroudsburg, PA: ACL, 2022: 3510−3520

[8] Zheng Mengxin, Lou Qian, Lei Jiang. Primer: Fast private transformer inference on encrypted data[J]. arXiv preprint, arXiv: 2303.13679, 2023

[9] Gupta K, Jawalkar N, Mukherjee A, et al. Sigma: Secure gpt inference with function secret sharing[J]. Cryptology ePrint Archive, 2023

[10] Juvekar C, Vaikuntanathan V, Chandrakasan A. GAZELLE: A low latency framework for secure neural network inference[C]//Proc of the 27th USENIX Conf on Security Symp. Berkeley, CA: USENIX Association, 2018: 1651−1669

[11] Jiang Xiaoqian, Kim M, Lauter K, et al. Secure outsourced matrix computation and application to neural networks[C]//Proc of the 2018 ACM SIGSAC Conf on Computer and Communications Security. New York: Association for Computing Machinery, 2018: 1209−1222

[12] Mohassel P, Zhang Y. SecureML: A system for scalable privacy-preserving machine learning[C]//Proc of 2017 IEEE Symp on Security and Privacy (SP). Piscataway, NJ: IEEE, 2017: 19−38

[13] Huang Zhicong, Lu Wenjie, Hong Cheng, et al. Cheetah: Lean and

fast secure two-party deep neural network inference[C]//Proc of the 31st USENIX Security Symp (USENIX Security 22). Berkeley, CA: USENIX Association, 2022: 809−826

[14] Wang Ning, Xiao Xiaohui, Yang Yin, et al. Collecting and analyzing multidimensional data with local differential privacy[C]//Proc of the 2019 IEEE 35th Int Conf on Data Engineering (ICDE). Piscataway, NJ: IEEE, 2019: 638−649

[15] Truong J B, Gallagher W, Guo Tian, et al. Memory-efficient deep learning inference in trusted execution environments[C]//Proc of the 2021 IEEE Int Conf on Cloud Engineering (IC2E). Piscataway, NJ: IEEE, 2021: 161−167

[16] Akavia A, Leibovich M, Resheff Y S, et al. Privacy-preserving decision trees training and prediction[J]. ACM Transactions on Privacy and Security, 2022, 25(3): 1−30

[17] Park S, Byun J, Lee J. Privacy-preserving fair learning of support vector machine with homomorphic encryption[C]//Proc of the ACM Web Conf 2022. New York: ACM, 2022: 3572−3583

[18] Mohassel P, Rindal P. ABY3: A mixed protocol framework for machine learning[C]//Proc of the 2018 ACM SIGSAC Conf on Computer and Communications Security. New York: ACM, 2018: 35−52

[19] Rathee D, Rathee M, Kumar N, et al. Cryptflow2: Practical 2-party secure inference[C]//Proc of the 2020 ACM SIGSAC Conf on Computer and Communications Security. New York: ACM, 2020: 325−342

[20] Hou Xiaoyang, Liu Jian, Li Jingyu, et al. Ciphergpt: Secure two-party gpt inference[J]. Cryptology ePrint Archive, 2023

[21] Li Dacheng, Shao Rulin, Wang Hongyi, et al. MPCFormer: Fast, performant and private transformer inference with MPC[J]. arXiv preprint, arXiv: 2211.01452, 2022

[22] Zeng Wenxuan, Li Meng, Xiong Wenjie, et al. MPCViT: Searching for MPC-friendly vision transformer with heterogeneous attention[J]. arXiv preprint, arXiv: 2211.13955, 2022

[23] Mishra P, Lehmkuhl R, Srinivasan A, et al. Delphi: A cryptographic inference system for neural networks[C]//Proc of the 29th USENIX Conf on Security Symp. Berkeley, CA: USENIX Association, 2020: 2505−2522

[24] Wang Xiaolong, Girshick R, Gupta A, et al. Non-local neural networks[C]//Proc of the IEEE Conf on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2018: 7794−7803

[25] Wang Sinong, Li B Z, Khabsa M, et al. Linformer: Self-attention with linear complexity[J]. arXiv preprint, arXiv: 2006.04768, 2020

[26] Rathee D, Rathee M, Goli R K K, et al. Sirnn: A math library for secure RNN inference[C]//Proc of 2021 IEEE Symp on Security and Privacy (SP). Piscataway, NJ: IEEE, 2021: 1003−1020

[27] Dong Ye, Lu Wenjie, Zheng Yancheng, et al. Puma: Secure inference of LLaMA-7B in five minutes[J]. arXiv preprint, arXiv: 2307.12533, 2023

[28] Akimoto Y, Fukuchi K, Akimoto Y, et al. Privformer: Privacy-preserving transformer with MPC[C]//Proc of 2023 IEEE 8th European Symp on Security and Privacy (EuroS&P). Piscataway, NJ: IEEE, 2023: 392−410

[29] Wagh S, Tople S, Benhamouda F, et al. Falcon: Honest-majority maliciously secure framework for private deep learning[J]. arXiv preprint, arXiv: 2004.02229, 2020

[30] Chou E, Beal J, Levy D, et al. Faster Cryptonets: Leveraging sparsity for real-world encrypted inference[J]. arXiv preprint, arXiv: 1811.09953, 2018

[31] Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network[J]. arXiv preprint , arXiv: 1503.02531, 2015

[32] Ma Junming, Zheng Yancheng, Feng Jun, et al. SecretFlow-SPU: A performant and user-friendly framework for privacy-preserving machine learning[C]//Proc of 2023 USENIX Annual Technical Conf (USENIX ATC 23). Berkeley, CA: USENIX Associatioin, 2023: 17−33

Liu Weixin, born in 2001. Master candidate. His main research interests include privacy-preserving machine learning, cryptography.

刘伟欣, 2001 年生. 硕士研究生. 主要研究方向为隐私保护机器学习、密码学.

Guan Yewei, born in 1999. PhD candidate. His main research interests include secure multi-party computation and privacy-preserving machine learning.

管晔玮, 1999 年生. 博士研究生. 主要研究方向为安全多方计算、隐私保护机器学习.

Huo Jiarong, born in 2002. Undergraduate. His main research interest includes applied cryptography.

霍嘉荣, 2002 年生. 本科生. 主要研究方向为应用密码学.

Ding Yuanchao, born in 1999. Master candidate. His main research interests include privacy-preserving machine learning and cryptography.

丁元朝, 1999 年生. 硕士研究生. 主要研究方向为隐私保护机器学习、密码学.

Guo Hua, born in 1980. PhD, associate professor. Member of CCF. Her main research interests include privacy-preserving machine learning and cryptography.

郭　华, 1980 年生. 博士, 副教授. CCF 会员. 主要研究方向为隐私保护机器学习、密码学.

Li Bo, born in 1981. PhD, associate professor. Member of CCF. His main research interests include privacy-preserving machine learning and network security.

李　博, 1981 年生. 博士, 副教授. CCF 会员. 主要研究方向为隐私保护机器学习、网络安全.