计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.241000041

基于信息融合的结构化剪枝算法研究

黄海新[1]　徐成龙[1]　付　垚[2]
1 沈阳理工大学自动化与电气工程学院　沈阳 110159
2 沈阳理工大学信息科学与工程学院　沈阳 110159

摘　要　针对现有大型语言模型在经过剪枝算法处理后在 Zero-shot Performance 中 PPL(困惑度)高、文本生成精度低、模型推理速度慢等问题,提出了一种基于损失联合量级为核心的剪枝度量算法(Loss And Magnitude,LAM)。在对权重重要性估计过程中,将损失函数信息、权重激活信息进行信息融合,使用 LAM 算法消除在权重重要性评估过程中对梯度信息进行 Taylor 展开时为提高计算效率省略二阶导数所造成的局限性,提高模型剪枝过程的准确率和鲁棒性,增强剪枝算法的泛用性。在建立耦合结构时,提出单向耦合结构,选择激活 Transformer 块中的多层感知机(MLP)中的神经元作为初始触发器,只需考虑向注意力层,查询向量、键向量、值向量层方向激活神经元建立耦合结构,从而降低了识别耦合结构组所需的参数量,提高剪枝速度和吞吐量。在 WikiText2 数据集和 PTB 数据集进行的 Zero-shot Performance 实验表明:在剪枝率为 25%时对 LLaMA-7B 进行剪枝处理,其 PPL 分数分别为 20.24 和 36.05,显著低于其他剪枝算法,在对 Vicuna-7B 剪枝后的 PPL 分数为 21.24 与 85.81,也优于其他剪枝算法,表现出更高的泛用性和准确性。

关键词:大语言模型;模型剪枝;Taylor+重要性估计;LoRa
中图分类号　TP391

Research on Structured Pruning Algorithm Based on Information Fusion

HUANG Haixin[1],XU Chenglong[1] and FU Yao[2]
1 School of Automation and Electrical Engineering,Shenyang Ligong University,Shenyang 110159,China
2 School of Information Science and Engineering,Shenyang Ligong University,Shenyang 110159,China

Abstract　Aiming at the problems of high PPL(perplexity),low text generation accuracy and slow model reasoning speed in Zero-shot Performance after the existing large-scale language model is processed by pruning algorithm,this paper proposes a pruning metric algorithm LAM based on the joint magnitude of loss. In the process of estimating the weight importance,the loss function information and the weight activation information are fused. By using the LAM algorithm,the limitations caused by the omission of the second derivative in the Taylor expansion of the gradient information in the process of weight importance evaluation are eliminated,and the accuracy and robustness of the model pruning process are improved. Enhance the versatility of the pruning algorithm. When establishing the coupling structure,a single coupling structure is proposed,and the neurons in the multi-layer perceptron(MLP) in the Transformer block are selected as the initial trigger. Only the attention layer,the query vector,the key vector,and the value vector layer are considered to activate the neurons to establish the coupling structure. Thus,the number of parameters required to identify the coupling structure group is reduced,and the pruning speed and throughput are improved. The Zero-shot Performance experiments on WikiText2 dataset and PTB dataset show that when the pruning rate is 25 %,the PPL scores of LLaMA-7B are 20.24 and 36.05,respectively,which are lower than other pruning algorithms. The PPL scores of Vicuna-7B after pruning are 21.24 and 85.81,which are also better than other pruning algorithms,showing that the algorithm has higher universality and accuracy.

Keywords　Large language models,Model pruning,Taylor+ importance estimates,LoRa

1　引言

基于 Transformer 的大型语言模型(Large Language Models,LLMs)在语言理解、文本生成和机器翻译等领域都表现出了显著的能力。LLMs 不仅能高质量完成自然语言生成任务,生成流畅通顺、贴合人类需求的语言,而且具备以生成式框架完成各种开放域自然语言理解任务的能力,而且在少样本、零样本场景下,大模型可取得接近乃至达到传统监督学习方法的性能,且具有较强的领域泛化性。但是,其庞大的参数数量和复杂的结构,导致大语言模型在训练和部署时需要耗费大量的计算资源和时间。因此,对大语言模型进行压缩处理已经成为目前亟需解决的问题。图 1 给出了 Transformer 结构图。

在模型压缩领域中常用的方法包括量化、知识蒸馏、模型剪枝,低秩分解等。其中模型剪枝作为优化深度学习模型的关键技术,通过剔除模型中“不重要”的权重或神经元,来减少模型的参数数量和计算量。其主要目的是在尽量保证模型精度的同时,降低模型的复杂度,提高计算效率,减少存储需求,

通信作者:黄海新(huanghaixin@sylu.edu.cn)

从而降低模型部署成本。

近年来，模型剪枝在大语言模型压缩工作中取得了一定的成果，其中 SparseGPT[1] 首先提出了一种不需要再培训的一次性修剪策略，即将剪枝视为一个广义稀疏回归问题，并使用近似稀疏回归求解器对其进行求解的思路对 LLMs 进行剪枝处理。Wanda[2] 引入了一种新的修剪度量，使用一小组校准数据对每个权重的大小和相应输入激活的范数的乘积来进行重要性评估，并在线性层的每个输出中对局部权重进行比较，去除较低优先级权重。LLMPruner[3] 基于梯度信息选择性去除非关键耦合结构，减少参数量，并通过 LoRA 技术快速恢复修剪模型的性能，进而最大限度地保留 LLM 的大部分功能。Sheared LLM[4] 使用端到端方式去除层、头、中间和隐藏维度，将更大的模型剪枝到指定的目标形状，并通过动态批加载算法，基于不同域的不同损失，动态更新每个训练批中采样数据的组成，恢复 LLMs 的性能。

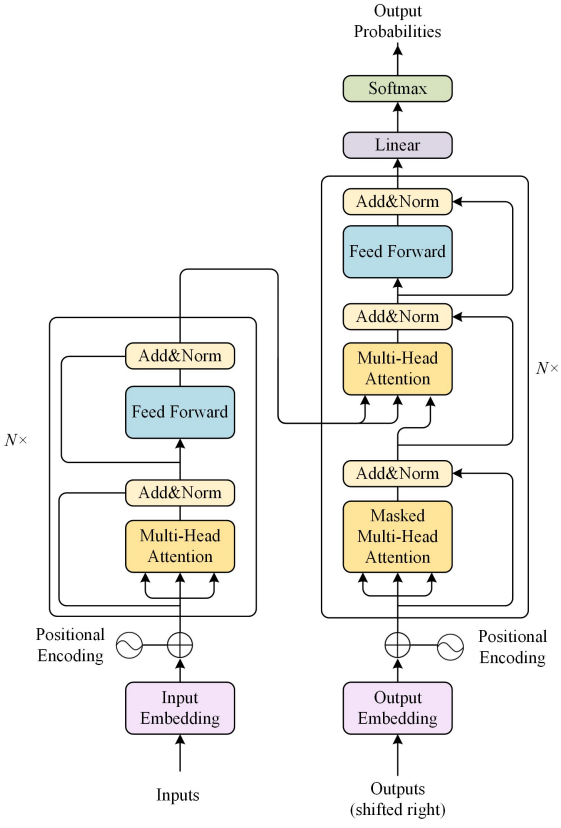

图 1 Transformer 结构图

Fig. 1 Structure diagram of transformer

但现有剪枝算法在对模型进行剪枝处理后，在 zero-shot 任务中存在性能较低和高剪枝率下的表现不佳等问题，为此，本文提出了一种基于损失联合量级为剪枝度量的算法(LAM)，将损失函数信息、权重激活信息进行信息融合，使用 LAM 算法消除在使用重要性评估指标 Taylor 时为提高计算效率省略二阶导数所造成的局限性，以提高模型剪枝过程的准确率和鲁棒性，增强剪枝算法的泛用性。在建立依赖结构时，提出单向耦合结构，选择激活 Transformer 块中的多层感知机(MLP)中的神经元作为初始触发器，而不是选择任意神经元作为初始触发器。此外，提出一种单向耦合结构，即只需考虑向注意力层，查询向量、键向量、值向量层方向激活神经元建立耦合结构，提高剪枝速度和吞吐量。

2 相关工作

2.1 Transformer 结构

大部分 LLMs 模型都采用 Transformer 模块，它是组成 LLMs 的基础单元，由多头注意力、前馈网络、Softmax、LayerNorm 等组成，其中注意力机制是针对一个文本序列，计算每个 token(符号)与其他 tokens 之间的相关系数，找出相关度高的 tokens，用于生成特征[5]。注意力机制是基于查询-键-值(QKV)计算的。具体算法为：输入文本序列 X_m，经过编码层对 X_m 进行编码，并添加位置信息，生成信息矩阵后与 $\boldsymbol{W}_Q$，$\boldsymbol{W}_K$，$\boldsymbol{W}_V$ 进行运算，生成 $\boldsymbol{QKV}$ 矩阵后，对 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$ 的转置进行矩阵乘法将查询和键矩阵的信息进行映射，除以 D_k，进行 Softmax 运算，得到注意力矩阵 $\boldsymbol{A}$；$\boldsymbol{A}$ 和 $\boldsymbol{V}$ 做矩阵乘，得到输出特征。计算式如式(1)所示：

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

其中，$\boldsymbol{Q}$，$\boldsymbol{K}$ 和 $\boldsymbol{V}$ 分别表示查询(Query)、键(Key)和值(Value)向量；$Attention$ 为自注意力机制的输出；softmax 为权重归一化函数；d_k 为 $\boldsymbol{K}$ 的维度；$1/\sqrt{d_k}$ 为缩放因子；$\boldsymbol{QK}^{\mathrm{T}}$ 是 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$ 之间的相似度。

前馈网络(FFN)由两个全连接层组成。经过第 1 个全连接层，特征维度由 D_m 扩大到 D_f；经过第 2 个全连接层，特征维度由 D_f 恢复到 D_m，计算式为如(2)所示：

$$FFN(H)=ReLU(H\,W^1+b^1)W^2+b^2 \tag{2}$$

其中，H 是本层输入，$W^1\in R^{D_m\times D_f}$，$W^2\in R^{D_f\times D_m}$，$b^1\in R^{D_f}$，$b^2\in R^{D_m}$。其中残差连接能够防止梯度消失，归一化层可以使特征数值维持在均值 0、方差 1。

2.2 模型剪枝原理

将 LLM 中的任意神经元视为初始触发器，它具有激活依赖于它的神经元的能力。随后，这些新触发的神经元可以作为后续的触发器来识别依赖并激活各自的依赖神经元。通过激活神经元，可以识别 LLM 中的相互存在依赖关系，并构建相互存在依赖关系的耦合结构组，重复激活神经元直到找到 LLM 中存在的所有耦合结构。初始触发器激活神经元如图 2 所示。

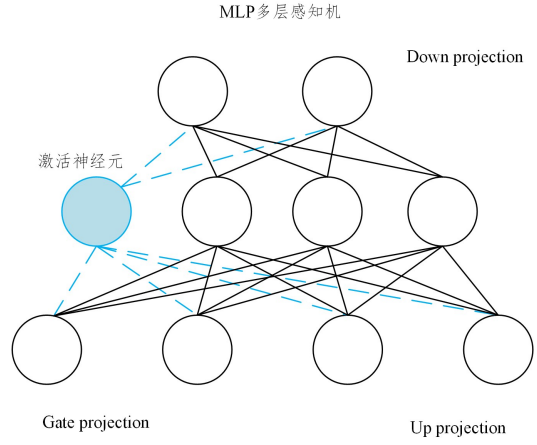

图 2 激活 MLP 的神经元

Fig. 2 Activate the neurons of the MLP

为了更准确地判断哪些结构需要被修剪，将所有的耦合结构进行分组处理，给定校正集 D，单个耦合结构组可以定义为 $G=\{W_i\}M_i=1$，其中 M_i 为一组耦合结构的数量，W_i 为每

个结构的权值。

在修剪时，我们的目标是去除对模型预测影响最小的组，单个权重W_i的重要性可以通过测量移除权重参数所引起的误差来量化，即通过 *Loss*（损失函数）中的偏差来表示。对模型内的所有耦合结构进行分组，计算每一个组的重要性分数，对各耦合结构组的重要性进行排序，并根据预定义的修剪比例修剪重要性较低的组。修剪 Transformer 中的耦合结构示意图如图 3 所示。

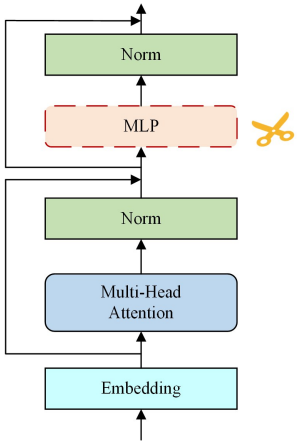

图 3　修剪 Transformer 中的耦合结构

Fig. 3　Trim the coupling structure in the Transformer

3　损失联合量级算法(LAM)

3.1　单向耦合结构

将 Transformer 块中的多层感知机（MLP）中的神经元设置为初始触发器，而不是将任意神经元作为初始触发器，通过建立向下的单向耦合结构，即只考虑向注意力头层，查询向量、键向量、值向量层方向激活神经元构建依赖结构组。这样做可以减少依赖检测算法向上层激活依赖神经元，降低了依赖算法检测结构组所需参数量，加快了剪枝速度和推理速度；并且，在 MLP 层设置初始触发器可以获得更完整的依赖结构组，提高模型生成文本的准确率。本文提出的单向耦合结构如图 4 所示。

3.1.1　依赖关系

建立将网络结构记为 $L=\{f_1,f_2,\cdots,f_L\}$ 其中 f 代表网络结构中的各种结构层，f_1 层中的输入输出，分别用 f_1^- 与 f_1^+ 表示。对于任意网络，最终分解可表示为 $L=\{f_1^-,f_1^+\cdots,f_L^-,f_L^+\}$，其中层间依赖关系和层内依赖关系可表示（$f_1^-$，$f_1^+$）$\Leftrightarrow$（$f_2^-$，$f_2^+$）与 $f_1^-\Leftrightarrow f_1^+$[6]。网络结构层的依赖模型如下：

$$D(f_i^-,f_j^+)=[f_i^-\leftrightarrow f_j^+]\vee[i=j\wedge sch(f_i^-)=sch(f_j^+)] \tag{3}$$

其中，“$\vee$”和“$\wedge$”表示“或”和“与”操作，$f_i^-\leftrightarrow f_j^+$ 表示依赖关系 $f_i^-\Leftrightarrow f_j^+$ 始终出现在结构层中，$sch(f_i^-)=sch(f_j^+)$ 表示为 f_i^- 和 f_j^+ 有相同的修剪方式。

在层内建立依赖关系时需要同时修剪单个层的输入和输出，如果 f_i^-，f_i^+ 共享相同的修剪方案，代表着在层内 f_i^-，f_i^+ 存在依赖关系，表示为 $sch(f_i^-)=sch(f_i^+)$。许多网络层都满足这个条件，例如批归一化，其输入和输出共享相同的修剪方案，因此将同时进行修剪。如果网络的拓扑结构已知，层间的依赖关系可以很容易地估计出来。

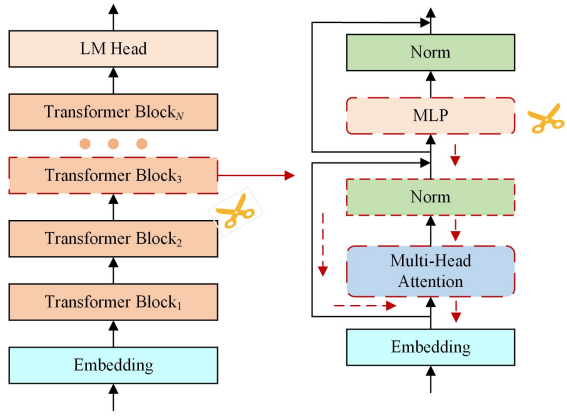

图 4　单向耦合结构剪枝

Fig. 4　Single-term coupling structure pruning

在神经元层级建立依赖关系，假设 N_i 和 N_j 是模型中的两个神经元，$In(N_i)$ 和 $Out(N_i)$ 表示指向 N_i 或指出 N_i 的所有神经元。结构间的依赖关系可以定义为：

$$N_j\in Out(N_i)\wedge Deg^-(N_j)=1\Rightarrow N_j\text{依赖于}N_i \tag{4}$$

其中，$Deg^-(N_j)$ 表示神经元 N_j 的 In 度。这种依赖关系是定向的，因此我们可以得到另一种依赖关系：

$$N_i\in \mathrm{In}(N_j)\wedge Deg^+(N_i)=1\Rightarrow N_i\text{依赖于}N_j \tag{5}$$

其中，$Deg^+(N_i)$ 表示神经元 N_i 的出度。这里的依赖原则表示为：如果当前神经元（例如 N_i）完全依赖于另一个神经元（例如 N_j），并且神经元 N_j 被修剪时，那么神经元 N_i 也必须进行修剪。

3.2　损失联合量级

对 LLMs 模型中所有存在依赖结构的神经元进行分组。为了评估每一组依赖结构组在 LLMs 模型中的贡献程度的高低，需要对结构组进行重要性评估，由于 LLMs 模型在预训练过程中没有开源训练所用的数据集，因此假设给定校准数据集为 $D=\{x_i,y_i\}_{i=1}^N$，其中 N 为样本数。单个耦合结构组的权重可以定义为 $W_A=\sum_{i=1}^{M}W_i$，其中 M 表示耦合结构组中依赖结构的数量，W_i 为每个结构的权值。

修剪的目标是去除对模型预测影响最小的组，权重 W_i 的重要性可以通过测量移除权重参数所引起的误差来量化，对误差进行 Taylor[7] 展开，通过 *Loss* 中的偏差变化估算权重 W_i 的重要性。

W_i 的重要性可以表示为：

$$\begin{aligned}\zeta_{W_i}&=|\Delta\zeta(D)|\\&=|\zeta_{W_i}(D)-\zeta_{W_0}(D)|\\&=\left|\frac{\partial\zeta^{\mathrm{T}}(D)}{\partial W_i}W_i-\frac{1}{2}W_i^{\mathrm{T}}\boldsymbol{H}W_i+o(\|W_i\|^3)\right|\end{aligned} \tag{6}$$

其中，$\boldsymbol{H}$ 是黑塞矩阵，在实际情况下为了减少计算量对二阶导数进行忽略处理。在耦合结构组中权重 w_i 的每一个参数的重要性估计可表示为：

$$\zeta_{W_i^k}=|\zeta_{W_i^k}(D)-\zeta_{W_i^k=0}(D)|=\left|\frac{\partial\zeta^{\mathrm{T}}(D)}{\partial W_i^k}W_i^k\right| \tag{7}$$

其中，k 表示权重 w_i 中第 k 个参数。但 Dettmers[8] 在对 LLM 进行量化处理时发现，在模型中存在一小部分隐藏状态特征的幅度（异常值）明显大于其他特征，并且将这些特征进行剪枝会导致性能显著下降。因此，在对权重重要性进行评估过程中，依赖组中权重 W_i 的大小（Magnitude）[9] 也会对模型预

测产生影响，并且权重W_i越大产生的影响就越大。为了更加准确地对模型权重进行剪枝处理，本文将权重梯度信息、量级信息及输入激活函数3个关键信息进行加权融合，形成了一个新的权重重要性评分，即损失联合量级(LAM)评价指标。使用LAM算法消除在使用重要性评估指标Taylor时为提高计算效率省略二阶导数所造成的局限性，并且LAM算法作为对关键信息加权融合的特征映射评估标准比只依靠梯度信息的评价指标更加敏感，可以对权重的重要性进行更加细致的评估，降低剪枝过程中的误检率，最大程度确保原模型性能。

因此需要修剪的A依赖结构组的重要性可以表示为：

$$\zeta_{W_A}=\varepsilon\sum_{i=1}^{M}\sum_{k}\zeta_{w_i^k}+\eta\sum_{i=1}^{M}\sum_{k}|w_i^kX_i^k| \tag{8}$$

其中，ε与η为平衡系数，X_i^k为权重W_i^k的输入激活函数。

4 模型再训练

使用量化低秩近似LoRA[10]技术对修剪后的模型进行后训练。模型中每个可学习的权重矩阵记为$\boldsymbol{W}$，其包含LLM中已修剪和未修剪的线性投影。$\boldsymbol{W}$的更新值$\Delta\boldsymbol{H}$可以分解为$\Delta\boldsymbol{W}=\boldsymbol{PQ}\in R^{d^-\times d^+}$，其中$\boldsymbol{P}\in\mathrm{R}^{\mathrm{d}^-\times\mathrm{d}}$，$\boldsymbol{Q}\in R^{d\times d^+}$适应网络的权值矩阵可表示为：

$$f(x)=(\boldsymbol{W}+\Delta\boldsymbol{W})\boldsymbol{X}+b=(\boldsymbol{WX}+b)+(\boldsymbol{PQ})\boldsymbol{X} \tag{9}$$

其中，b是密集层中的偏置。再训练过程中只训练$\boldsymbol{P}$和$\boldsymbol{Q}$降低了整体的训练复杂度，减少了对大规模训练数据的需求。另外，额外的参数$\boldsymbol{P}$和$\boldsymbol{Q}$可以重新参数化为$\Delta\boldsymbol{W}$，不会在最终的压缩模型中产生额外的参数。

5 实验与结果分析

5.1 评估指标

Zero-shot Performance通常用于描述模型在未经过针对特定任务的训练或微调的情况下的性能，本文使用PPL语言困惑度作为Zero-shot Performance评价指标，在自然语言处理任务中PPL(Perplexity)[11]是指语言模型对于给定测试集的困惑度，即模型对于输入序列的预测不确信程度。PPL评价指标在自然语言处理领域应用广泛，主要用来衡量模型的好坏。一般来说，PPL值越小，说明模型对于输入序列的预测能力越强，模型表现越好。在具体实现中，PPL计算式如下：

$$PPL=\exp(-1/N\Sigma\log\Pr(w|prev\_w)) \tag{10}$$

其中，$\Pr(w|prev_w)$表示在给定前一个词的情况下，模型预测下一个词的概率；N表示输入序列的长度。

在大语言模型(LLMs)中Throughput[12](吞吐量)是一个关键的性能指标，它衡量的是模型在单位时间内能够处理的输入数据量，通常以每秒可以处理的样本数(samples/s)或者每秒可以处理的tokens数(tokens/s)来表示。高吞吐量意味着模型可以在相同的时间内处理更多的数据，代表模型有着更高的的资源利用率和成本效率。

5.2 实现细节

本实验在网络改进及训练、测试过程中所用的操作系统为Windows系统，CPU型号为Intel(R) Core(TM) i5 G12500H，GPU处理器为NvidiaRTX3090，GPU加速库CUDA11.8，并在基于Python语言和PyTorch1.7.1框架搭建的深度学习环境下进行。

设定校正集BookCorpus[13]样本数N为10，在对剪枝模型进行LoRA再训练时，使用alpaca-cleaned[14]数据集中的50000条文本进行再训练，训练轮次为5轮，batch-size设置为64，初始学习率设置为0.0001，lora_r设置为8，lora_alpha为16，lora_dropout设置为0.05，ε为1.25，η为0.5。

5.3 实验与结果分析

为了探究改进后的剪枝算法是否在效果上有提升，以及是否有较好的泛用性，本文将所提出的剪枝算法与当前主流剪枝算法进行了对比实验，结果如表1、表2所列。

表1 本文算法与其他剪枝算法在LLaMA-7B的指标对比

Table 1 Comparison between the proposed algorithm and other pruning algorithms in LLaMA-7B

Pruning Ratio	Method	Param/B	WikiText2↓	PTB↓	Thr↑ (tokens/s)
Ratio=0%	LLaMA-7B	6.7	12.60	22.10	25.0
Ratio=20%	Wanda-sp	5.4	22.40	36.15	16.9
	FLAP		17.00	**30.10**	16.5
	LLMPruner		17.38	30.11	22.1
	LAM		**16.93**	31.86	**24.2**
Ratio=25%	Wanda-sp	4.9	25.53	40.51	16.3
	FLAP		21.34	37.12	15.7
	LLMPruner		20.35	36.17	20.4
	LAM		**20.24**	**36.05**	**23.1**
Ratio=30%	Wanda-sp	4.7	27.08	46.25	16.4
	FLAP		23.24	40.40	15.4
	LLMPruner		22.40	**39.70**	21.3
	LAM		**23.48**	53.57	**22.4**
Ratio=35%	Wanda-sp	4.5	31.31	62.36	16.1
	FLAP		26.65	44.36	15.8
	LLMPruner		**25.20**	**40.73**	**21.6**
	LAM		29.11	59.46	21.2

表2 本文算法与其他剪枝算法在Vicuna-7B的指标对比

Table 2 Comparison between the proposed algorithm and other pruning algorithms in Vicuna-7B

Pruning Ratio	Method	Param/B	WikiText2↓	PTB↓	Thr↑ (tokens/s)
Ratio=0%	Vicuna-7B	6.7	17.10	63.20	25.0
Ratio=20%	Wanda-sp	5.4	24.41	94.60	16.3
	FLAP		22.40	**74.91**	16.6
	LLMPruner		**19.69**	78.43	21.4
	LAM		20.93	75.20	**24.2**
Ratio=25%	Wanda-sp	4.9	33.50	113.2	15.4
	FLAP		24.19	86.24	15.7
	LLMPruner		21.70	85.90	22.9
	LAM		**21.24**	**85.81**	**23.5**
Ratio=30%	Wanda-sp	4.7	60.40	146.9	15.2
	FLAP		27.12	95.40	15.3
	LLMPruner		**25.59**	94.10	21.7
	LAM		25.74	**91.53**	**22.1**
Ratio=35%	Wanda-sp	4.5	73.20	186.5	16.1
	FLAP		34.60	104.8	15.8
	LLMPruner		**27.60**	**102.2**	21.1
	LAM		29.74	111.7	**21.8**

在对LLaMA-7B[15]与Vicuna-7B[16]进行剪枝处理时，当剪枝率为20%时，LLaMA2-7B剪枝后模型参数剩余量为5.4B，在数据集WikiText2与PTB进行的Zero-shot Performance中的PPL分值分别为16.93和31.86，仅在PTB上的困惑度得分低于其他剪枝算法。

当剪枝率为25%时，LLaMA-7B的模型参数量为4.9B，其PPL分值分别为20.24和36.05，明显低于其他剪枝算法，证明本文提出的剪枝算法在对LLaMA-7B模型修剪率为25%的条件下，本算法保留了原模型大部分的性能，在文本生

成性能上优于其他算法。

但在剪枝率为35%时在数据集 WikiText2 的 Zero-shot 性能任务中的 PPL 分值仅优于 Wanda-sp[17] 算法，远高于 FLAP[18] 与 LLMPruner 算法，这表明改进后的剪枝算法在高剪枝率下的大语言模型出现了较为明显的性能下降，这将成为未来需要改进的方向。为了检验本文提出的剪枝算法 LAM 的泛用性，将 Vicuna 模型作为对照模型，进行剪枝处理。Vicuna 模型是一个开源的对话生成模型，旨在提供高质量的自然语言理解和生成能力。它在 LLAMA 模型的基础上进行了微调，相比 LLAMA 模型更适用于对话系统和人机交互场景，专注于提高对话的连贯性和上下文理解。在 Vicuna-7B 模型剪枝中，也产生了与 LLaMA-7B 模型相似的结果，可以看到，在剪枝率为25%时本文算法（LAM）优于其他主流方法，但在较高剪枝率下，如剪枝率为35%时，也出现了明显的性能下降问题。LAM 算法的 Thr（Throughput）吞吐量相比其他剪枝算法有更好的表现，表明改进后的算法有较高的计算效率。但与原模型相比，剪枝后的模型即使修剪了模型参数及内部复杂的网络结构后，其吞吐量非但没有上升反而出现下降。这是因为，剪枝后的模型通常需要进行微调以恢复因剪枝而损失的性能。但是，由于只对剪枝后的50000条文本进行微调训练，导致模型的性能无法恢复到预期水平，从而导致吞吐量下降。

对剪枝后模型进行 LoRA 再训练的损失函数变化图，如图5所示。

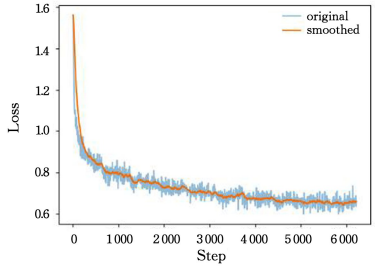

图5　LoRA 训练时损失函数图

Fig. 5　LoRA loss function plot during training

5.4　消融实验

为了验证本文使用的损失联合量级（LAM）重要性评估这一策略的有效性，本节在 LLaMA-7B 上做了消融实验，结果如表3所列，其中 Magnitude -L1[19]，Magnitude-L2[20]，Taylor 均为主流的重要性评估准则。

表3　消融实验结果

Table 3　Ablation experiments results

Pruning Ratio	Method	WikiText2 ↓	PTB ↓
Ratio=0%	LLaMA-7B	17.10	63.20
Ratio=20%	Magnitude-L1	267.43	596.29
	Magnitude-L2	227.18	526.57
	Taylor	20.20	35.30
	LAM	**19.24**	**34.86**
Ratio=25%	Magnitude-L1	370.46	642.78
	Magnitude-L2	356.74	623.52
	Taylor	25.18	39.53
	LAM	**23.54**	**38.67**
Ratio=30%	Magnitude-L1	496.22	731.98
	Magnitude-L2	441.49	718.45
	Taylor	29.90	42.00
	LAM	**26.35**	**41.72**

Magnitude-L1 和 Magnitude-L2 方案表示除使用 L1 和 L2 范数作为重要性评估基准之外，其余策略与本文方法一致的剪枝方案，Taylor 方案则是仅使用 Taylor 一阶展开式外，其他与本文采用相同的剪枝方案。

当剪叶率达到20%时，依赖于量级的算法开始崩溃，导致信息丢失。Taylor 准则虽然提高了在文本生成任务的准确性，但依然出现性能迅速下降的问题。LAM 算法相比其他算法，可以在保留相同参数量的条件下获得更高的文本生成质量。

从整体上来看，在没有对剪枝模型进行恢复性微调的条件下，本文提出的方法在剪枝率为20%时，明显优于采用 L1，L2 范数的剪枝方案以及 Taylor 剪枝方案，在剪枝率为30%时，LAM 算法虽然出现了轻微的性能下降，但相较于其他算法，LAM 算法仍然存在优势。这表明使用基于信息融合的 LAM 算法可以对权重重要性有更加细致的评估，相比其他剪枝算法，能够更好地保证模型性能。

结束语　针对大型语言模型庞大的参数和复杂的结构需要大量的计算资源和时间，以及现有大语言模型在经过剪枝算法处理后在 Zero-shot Performance 表现中困惑度高、精度低、模型推理速度慢等问题，本研究提出了一种基于损失联合量级为核心的剪枝度量算法 LAM。使用将梯度、Magnitude 与输入激活函数这3个重要指标进行信息融合的修剪度量作为估计每个耦合结构组对模型的重要性大小，并提出了单向耦合结构，减少参数计算量。大量对比实验证明了相较于其他目前主流的模型剪枝算法，本文提出的剪枝算法在准确率与吞吐量上优于其他剪枝算法，表现出较好的泛用性与鲁棒性。但本文提出的算法还存在一些缺点，例如，算法在较高剪枝率的 Zero-shot Performance 表现较弱，未来计划通过实验来进一步探索如何提高较高剪枝率下 Zero-shot 任务表现能力。

参 考 文 献

[1] FRANTAR E, ALISTARH D. Sparsegpt: Massive language models can be accurately pruned in one-shot[C]// International Conference on Machine Learning. PMLR, 2023: 10323-10337.

[2] SUN M, LIU Z, BAI R A, et al. A simple and effective pruning approach for large language models[J]. arXiv: 2306.11695, 2023.

[3] MA X, FANG G, WANG X. Llm-pruner: On the structural pruning of large language models[J]. Advances in Neural Information Processing Systems, 2023, 36: 21702-21720.

[4] KIM B K, KIM G, KIM T H, et al. Shortened llama: A simple depth pruning for large language models[J]. arXiv: 2402.02834, 2024.

[5] ZHU X P, YAO H D, LIU J, et al. Review of Evolution of Large Language Model Algorithms[J]. ZTE Technology Journal, 2024, 30(2): 9-20.

[6] FANG G, MA X, SONG M, et al. Depgraph: Towards any structural pruning[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023: 16091-16101.

[7] MOLCHANOV P, MALLYA A, TYREE S, et al. Importance

estimation for neural network pruning[C]// CVPR. 2019.
[8] DETTMERS T, LEWIS M, BELKADA Y, et al. LLM. int8(): 8-bit matrix multiplication for transformersat scale[J]. arXiv: 2208.07339, 2022.
[9] LI H, KADAV A, DURDANOVIC I, et al. Pruning filters for efficient convnets[C]// ICLR. 2017.
[10] HU E J, SHEN Y, WALLIS P, et al. Lora: Low-rank adaptation of large language models[J]. arXiv: 2106.09685, 2021.
[11] HE T W, WANG H. Evaluating Perplexity of Chinese Sentences Based on Grammar & Semantics Analysis[J]. ApplicationResearch of Computers, 2017, 34(12): 3538-3542, 3546.
[12] KIM B K, KIM G, KIM T H, et al. Shortened llama: A simple depth pruning for large language models[J]. arXiv: 2402.02834, 2024.
[13] ZHU Y K, KIROS R, ZEMEL R, et al. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books[C]// ICCV. 2015.
[14] TAORI R, GULRAJANI I, ZHANG T Y, et al. Stanford alpaca: An instruction-following llama model [EB/OL]. https://github.com/tatsu-lab/stanford_alpaca.
[15] TOUVRON H, MARTIN L, STONE K, et al. Llama 2: Open foundation and fine-tuned chat models[J]. arXiv: 2307.09288, 2023.
[16] CHIANG W L, LI Z, LIN Z, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90% * chatgpt quality[J]. See https://vicuna.lmsys.org(accessed 14 April 2023), 2023, 2(3): 6.
[17] SUN M J, LIU Z, BAIR A, et al. A simple and effective pruning approach for large language models[C]// ICLR. 2024.
[18] AN Y, ZHAO X, YU T, et al. Fluctuation-based adaptive structured pruning for large language models[C]// Proceedings of the AAAI Conference on Artificial Intelligence. 2024: 10865-10873.
[19] LV B, ZHOU Q, DING X, et al. KVPruner: Structural Pruning for Faster and Memory-Efficient Large Language Models[J]. arXiv: 2409.11057, 2024.
[20] CHENG H, ZHANG M, SHI J Q. MINI-LLM: Memory-Efficient Structured Pruning for Large Language Models[J]. arXiv: 2407.11681, 2024.

HUANG Haixin, born in 1973, Ph. D, master's supervisor. Her main research interests include machine learning, artificial intelligence, and natural language processing.