中图法分类号:TP391.4　　**文献标识码**:A　　**文章编号**:1006-8961(2025)02-0485-18

论文引用格式:Qiu M B, Gao J, Lin S B, Li L, Wang G, Hu W M and Wang Y Z. 2025. Efficient Transformer tracking for the edge end with linearly decomposed attention. Journal of Image and Graphics, 30(02):0485-0502(邱淼波, 高晋, 林述波, 李椋, 王刚, 胡卫明, 王以政. 2025. 线性分解注意力的边缘端高效Transformer跟踪. 中国图象图形学报, 30(02):0485-0502)[DOI:10.11834/jig.240192]

线性分解注意力的边缘端高效Transformer跟踪

邱淼波[1,2],高晋[1,2*],林述波[1,2],李椋[3],王刚[3],胡卫明[1,2,4],王以政[3]

1. 中国科学院自动化研究所,北京　100190; 2. 中国科学院大学人工智能学院,北京　101408;
3. 军事医学研究院军事认知与脑科学研究所,北京　100850; 4. 上海科技大学信息科学与技术学院,上海　201210

摘　要:**目的**　将面向服务器端设计的跟踪算法迁移部署到边缘端能显著降低功耗,具有较高的实用价值。当前基于Transformer的跟踪算法具有明显的性能优势,然而部署在边缘端时,却可能产生较高的延迟。为了解决这个问题,提出了一种面向边缘端的线性分解注意力(linearly decomposed attention,LinDA)结构,可有效降低Transformer的计算量和推理延迟。**方法**　LinDA将多头注意力近似表示成数据依赖部分和数据无关部分的和:对于数据依赖部分,用简单的向量元素间相乘及求和表示,避免了复杂的转置和矩阵乘法;对于数据无关部分,直接利用统计得到的注意力矩阵,然后加上一个可学习偏置向量。这种分解既具有全局注意力,又保持了数据依赖的优点。为了弥补线性分解带来的精度损失,还设计了一种知识蒸馏方案,它在原始的损失函数上增加了两部分蒸馏损失:1)将真实包围框替换成教师模型预测的包围框作为监督目标,称为硬标签知识蒸馏;2)将教师模型预测得分的相对大小作为监督目标,称为关系匹配知识蒸馏。基于LinDA结构进一步实现了一种面向边缘端的目标跟踪算法LinDATrack,并将其部署在国产边缘计算主机HS240上。**结果**　在多个公开数据集上进行了评测。实验结果表明,该算法在该计算主机上可达到61.6帧/s的跟踪速度,功耗约79.5 W,功耗仅占服务器端的6.2%,同时其在LaSOT和LaSOT_ext上的成功率(success rate, SUC)相对于服务器端基线算法SwinTrack-T最多仅下降约1.8%。**结论**　LinDATrack具有良好的速度和精度平衡,在边缘端具有较大的优势。

关键词:目标跟踪;边缘端;Transformer;多头注意力(MHA);知识蒸馏

Efficient Transformer tracking for the edge end with linearly decomposed attention

Qiu Miaobo[1,2], Gao Jin[1,2*], Lin Shubo[1,2], Li Liang[3], Wang Gang[3], Hu Weiming[1,2,4], Wang Yizheng[3]

1. Institute of Automation, Chinese Academy of Sciences, Beijing　100190, China;
2. School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing　101408, China;
3. The Brain Science Center, Beijing Institute of Basic Medical Sciences, Beijing　100850, China;
4. School of Information Science and Technology, ShanghaiTech University, Shanghai　201210, China

Abstract: **Objective**　The transfer and deployment of tracking algorithms designed for server ends to edge ends has high

收稿日期:2024-04-07;**修回日期**:2024-06-17;**预印本日期**:2024-06-23

*通信作者:高晋　jin.gao@nlpr.ia.ac.cn

基金项目:科技创新2030—"新一代人工智能"重大项目(2020AAA0105802, 2020AAA0105800);国家自然科学基金项目(U22B2056, 62306312);北京市自然科学基金项目(JQ22014, L223003, 4234087);中国科学院青年创新促进会项目(2020142)

Supported by: National Key R&D Program of China (2020AAA0105802, 2020AAA0105800); National Natural Science Foundation of China (U22B2056, 62306312); Beijing Natural Science Foundation (JQ22014, L223003, 4234087); Youth Innovation Promotion Association of the Chinese Academy of Sciences (2020142)

practical value. This transformation leads to a remarkable decrease in energy consumption, particularly in situations where resources are limited. In recent years, tracking algorithms that incorporate the Transformer architecture have achieved considerable progress because of their superior performance. Nonetheless, the adaptation of these algorithms for edge computing often encounters difficulties, primarily because of the increased latency. This latency is attributed to the complex nature of the Transformer's attention mechanism, which requires extensive computational resources. This issue is addressed by introducing an innovative solution called the linearly decomposed attention (LinDA) module, which is designed expressly for edge computing. By drastically lowering the computational demands and reducing the inference time of the Transformer, the LinDA module facilitates more effective and efficient tracking at the edge end. **Method** LinDA innovatively approximates the multihead attention mechanism as two components: a data-dependent component and a data-independent component. For the data-dependent aspect, LinDA adopts a computationally economic approach. Rather than relying on traditional, resource-intensive methods of transposition and matrix multiplication, LinDA employs direct elementwise multiplication and the addition of vectors. This method markedly reduces computational complexity, rendering it exceptionally well suited for edge computing environments where resources are scarce. Regarding the data-independent facet, LinDA integrates a statistically derived attention matrix that encapsulates global contextual insights. This matrix is further refined with a learnable bias vector, enhancing the model's adaptability and versatility. This decomposition strategy empowers LinDA to achieve good precision and considerable efficiency on devices constrained by limited resources. An advanced knowledge distillation strategy, which plays a crucial role in bolstering the student model's capabilities, is introduced to mitigate potential compromises in accuracy because of the linear decomposition approach. This strategy encompasses two specialized distillation losses integrated into the baseline loss function, each meticulously designed to capture and convey critical insights from the teacher model to the student model. First, the hard label knowledge distillation technique involves replacing the ground-truth bounding box with the bounding box predicted by the teacher model, which serves as the supervision target for the student model. This method allows the student model to learn directly from the teacher's discernment, thereby enhancing its predictive precision. Consequently, the student model captures the teacher's knowledge of the problem, which enables it to yield more accurate predictions. Second, the relation matching knowledge distillation strategy harnesses the relationship between the teacher model's predictions as the supervisory target. This innovative approach captures the complex relationships among different predictions, such as the relative significance of distinct objects or their spatial interrelations. When this relational knowledge is embedded into the student model during training, the model's performance can be further improved, rendering it more robust and powerful. In summary, this elaborate knowledge distillation framework successfully imparts the teacher model's insights into the student model, effectively overcoming the potential precision degradation associated with linear decomposition. This scheme ensures that the student model inherits the teacher's expertise, thereby enabling it to deliver more precise predictions and attain superior performance. This study further implements an edge-end-oriented object tracking algorithm called LinDATrack, which is based on LinDA and distillation. The algorithm is deployed on the domestic edge computing host HS240. **Result** Comprehensive experiments of the tracker are conducted across various public datasets to test its performance and capability metrics. The experimental results validate the system's outstanding tracking speed and good precision. With this computing host, LinDATrack achieves an impressive tracking speed of approximately 62 frames per second, facilitating efficient tracking in real-time settings. Furthermore, the system operates with a power consumption of approximately 79.5 watts, which represents only 6.2% of the energy used by the server-end configurations. This dramatic reduction in energy usage underscores the system's exceptional energy efficiency, positioning it as an ideal choice for deployment in settings with limited resources. In an era where energy conservation and sustainability are increasingly important, this system presents a compelling alternative to more energy-intensive options, contributing significantly to a more sustainable computing landscape. In addition to its remarkable tracking speed and low power consumption, the system also exhibits consistently high tracking accuracy, distinguishing it within the realm of object tracking. Compared with the server-end baseline algorithm SwinTrack-T, the system's tracking accuracy, as determined by the success rate metric, shows only a slight decrease of approximately 1.8%. This minor decrease in accuracy reflects the system's capacity to balance performance with efficiency. The scheme can maintain precise tracking functionality while reducing resource usage, rendering it a versatile solution for a broad spec-

trum of tracking applications. **Conclusion** LinDATrack is distinguished by its exceptional balance of speed and accuracy, positioning it as a premier option for object tracking applications. Its performance is marked by efficiency, facilitating real-time tracking that users can rely on. Additionally, LinDATrack demonstrates considerable strengths when deployed at the edge, making it exceptionally well suited for environments with limited resources. This combination of speed, accuracy, and edge-oriented advantages firmly establishes LinDATrack as a leading solution for edge-end tracking tasks.

Key words: object tracking; edge end; Transformer; multi-head attention(MHA); knowledge distillation

0 引 言

随着 vision Transformer (ViT) (Dosovitskiy 等, 2021) 的提出,Transformer在目标跟踪中也得到了广泛应用。由于它基于令牌(token)的数据形式和具有多头注意力结构,使得基于孪生架构跟踪器的两个分支能很好地进行特征融合,因此在目标跟踪中迅速普及。然而,基于 Transformer 的跟踪器 SwinTrack(Lin 等,2022)部署在国产边缘计算主机 HS240(见附录A)上时存在一些问题,如部分算子不支持。

本文先对 SwinTrack 进行适当改进,使得能在 HS240 上成功部署,并将它作为基线模型。通过实验发现,基线模型中编码器(encoder)和解码器(decoder)的延迟占主要部分。这与多头注意力有关。多头注意力中主要包括注意力矩阵的计算、softmax 和特征重组等操作。本文进一步测试了多头注意力结构中各个部分的延迟贡献,发现延迟主要来自于注意力矩阵的计算,它由一个 reshape 操作、矩阵乘法和元素间乘法组成。而特征重组部分的计算复杂度虽然与它相同,但是延迟却很低。因此,多头注意力的延迟并非主要来自于单纯的矩阵乘法,而是主要来自于 reshape、矩阵乘法的组合。本文希望设计一种高效、轻量的多头注意力结构,从而降低计算量和推理延迟。

多头注意力的轻量化通常有两种研究路线,一种路线是线性化,通过简化注意力矩阵的计算过程来减少计算量,如 Performers(Choromanski 等,2022)和 SOFT(Lu 等,2021);另一种研究路线是稀疏化,通过减少令牌数来减少计算量,如 Sparse Transformer(Child 等,2019)和 Explicit Sparse Transformer(Zhao 等,2019)。这些轻量化工作通常是面向服务器端、精度优先。而本文的工作主要是面向边缘端、延迟优先(即速度优先)。

本文提出一种形式简单的线性分解注意力(linearly decomposed attention,LinDA)结构,并从数学上证明这种近似的可行性。LinDA 的核心思想是用两个独立随机变量的线性组合来近似表示它们的乘积,它将多头注意力表示成数据依赖部分和数据无关部分的和。这种分解既能保持全局注意力、数据依赖的优点,又能降低复杂度。由于最终结构的 LinDA 仅包含向量的元素间乘法、矩阵内沿着某个坐标轴求和,以及矩阵间求和,计算非常简单,因此具有较低的延迟。

单独使用 LinDA 会导致一定的精度损失,本文希望在保持模型结构的同时,能提高模型精度。知识蒸馏是一种较好的选择。提出基于硬标签和关系匹配的知识蒸馏来充分利用教师模型的知识。对于硬标签知识蒸馏,借鉴 DeiT(Touvron 等,2021),将教师模型预测的结果当做另一种监督目标并计算损失函数。对于关系匹配知识蒸馏,将教师模型预测的多个包围框与真实包围框的交并比(intersection over union,IoU)相对大小当做监督目标,计算教师模型和学生模型的Pearson 距离作为损失函数。

将模型部署在国产边缘计算主机 HS240上,并在多个公开数据集上进行测试,实验结果表明,能达到 61.6帧/s的推理速度,在 LaSOT上的成功率(success rate,SUC)达到 66.3%,超过众多服务器端或移动端最先进跟踪算法,并具有较低的功耗。

总的来说,本文主要贡献如下:1)设计了一种 LinDA结构,降低了多头注意力机制的延迟和计算复杂度,使得延迟降低、计算量减少,非常适合低功耗的边缘端部署;2)提出一种知识蒸馏方案,能有效地弥补线性分解的精度损失;3)提出一种边缘端跟踪器 LinDATrack,并部署在国产边缘计算主机 HS240上,速度达到61.6帧/s,功耗为79.5 W,功耗降低为服务器端的6.2%。

1 相关工作

1.1 目标跟踪中的Transformer

由于Transformer内在地具备融合多分支信息并进行信息交互的能力，因此，它首先应用于孪生结构跟踪器的特征融合阶段。TransT（Chen等，2021）设计了自上下文增强（ego-context augment，ECA）模块和交叉特征增强（cross-feature augment，CFA）模块，并用它们来做融合，使用交叉注意力代替相关操作。TrDiMP（Wang等，2021）使用Transformer增强模板特征以获得更加鲁棒的目标表示，并使用交叉注意力改进模板匹配过程，将有价值的跟踪线索从历史模板传递到当前帧。STARK（Yan等，2021a）使用编码器融合模板和搜索区域的特征，使用解码器更新模板，同时捕获时间和空间信息，并利用解码器直接预测包围框，无需后处理，简化跟踪过程。

Transformer具有良好的全局感受野及强大的特征表达能力，也可以作为主干网络来提取特征。SwinTrack（Lin等，2022）将SwinTransformer作为主干网络，在达到较高精度的同时，也能避免过高的计算量。一些研究者发现，在特征提取阶段同时进行特征融合对目标跟踪很重要。MixFormer（Cui等，2022）使用一组混合注意力模块（mixed attention module，MAM）来同时提取和集成目标模板及搜索区域的特征。MixFormer的精度虽然较高，但是跟踪速度较慢，这是因为MAM模块的计算效率很低。其在GTX 1080Ti GPU上速度是25帧/s。OSTrack（Ye等，2022）使用ViT作为主干网络来进行特征提取和特征融合。作者设计了一个早期候选消除模块，删除包含背景信息的令牌，减少计算量，提高了跟踪速度，同时减少背景的干扰，提高了精度。单流跟踪器虽然在精度上表现出色，但是需要在推理阶段提取模板特征，速度较慢，不利于在边缘端的部署和应用。

1.2 注意力机制的轻量化

降低注意力机制的复杂度通常有两种路线，一种是线性化。这种路线通常使用独立的核函数取代自注意力中的softmax函数。Performers（Choromanski等，2022）使用正交随机特征来近似softmax运算。T2R（Kasai等，2021）通过使用线性复杂度的递归取代softmax函数来减小时间和内存消耗。SOFT（Lu等，2021）首次提出一种没有softmax的Transformer，通过矩阵分解来近似自注意力矩阵。EfficientVit（Cai等，2023）使用深度卷积来提高线性注意力的局部特征提取能力。

另一条研究路线是稀疏化，其做法是减少每个令牌依赖的其他令牌的数量，从而减少计算量。Sparse Transformer（Child等，2019）将空洞自注意力和局部自注意力结合，提出Sparse self-attention，既能学习局部的特性，又能学习远程稀疏的相关性。与Sparse Transformer不同，Explicit Sparse Transformer（Zhao等，2019）先根据$\boldsymbol{Q}$和$\boldsymbol{K}$生成注意力得分矩阵$\boldsymbol{P}$，再根据$\boldsymbol{P}$选择相关性最大的k个元素。

1.3 知识蒸馏

知识蒸馏是一种将复杂模型的知识迁移到较小模型的经典方法。原始的知识蒸馏（Hinton等，2015）使用教师模型预测的logits作为学生模型的目标。此后，大量的变体被提出，如基于特征对齐、基于特征图或数据样本对之间的关系进行建模。Touvron等人（2021）引入一个蒸馏令牌，允许Transformer从卷积网络中学习。Hao等人（2022）利用图像和补丁之间的连接信息从教师Transformer中挖掘知识。Wu等人（2022）利用知识蒸馏，将较大模型的知识迁移到较小模型，同时使得较小模型能利用大量预训练数据的红利。

2 预备知识与基线分析

2.1 预备知识

2.1.1 目标跟踪框架

基于孪生网络的目标跟踪框架如图1所示。整个跟踪器由一个特征提取部分、特征融合部分和预测头组成。模板图像和搜索图像分别经过共享的特征提取网络得到特征图，特征图经过拉伸、拼接成合适的形状，依次输给编码器、解码器，得到响应图，响应图再输入给回归头和分类头，得到预测包围框和置信度，再经过后处理得到最终的预测包围框。编码器、解码器的结构见附录B。

如图1所示，特征提取与特征融合分开，整个模型包含两个分支，属于双流跟踪器；如果特征提取与特征融合同时进行，也就是模板分支（z分支）和搜索分支（x分支）在输入阶段进行拼接，此时整个模型只包含一个分支，属于单流跟踪器。对于特征融合

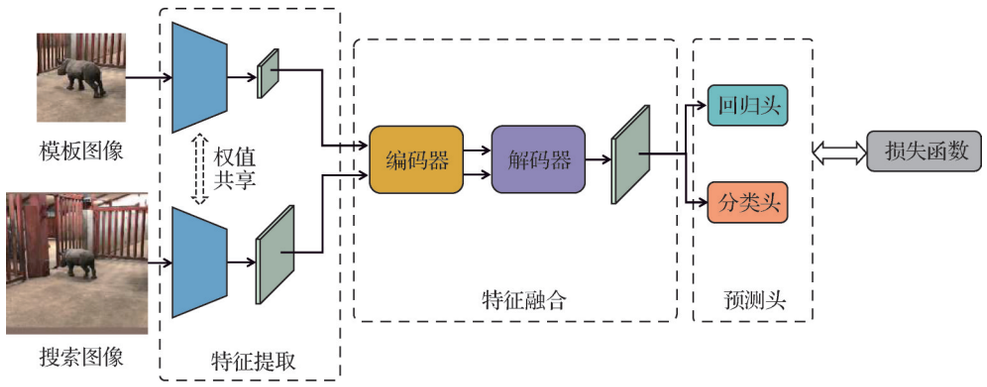


图 1　目标跟踪框架

Fig. 1　Pipeline of object tracking

部分，判别式跟踪器（Danelljan 等，2019；Bhat 等，2019；王法胜 等，2023）使用相关滤波做融合，早期的孪生跟踪器（Bertinetto 等，2016；Li 等，2018；Zhu 等，2018；尚欣茹 等，2021；陈志良和石繁槐，2022）使用相关操作进行融合。近期，也有一些工作（Lin 等，2022；Chen 等，2021；薛万利 等，2024）使用堆叠的 Transformer 层进行融合。可以先使用编码器，再使用解码器；也可以不使用解码器，而把编码器输出中的一部分作为特征融合的输出。普通的目标跟踪只有普通的分类损失和回归损失，本文在此基础上增加了两种蒸馏损失，将在 3.2 节详细介绍。

2.1.2　多头注意力结构

在标准的缩放点乘注意力（scaled dot-product attention）中，$\boldsymbol{AttMat}$ 和 $\boldsymbol{Attention}$ 分别为

$$\boldsymbol{AttMat} = \text{similarity}(\boldsymbol{Q}, \boldsymbol{K}) = \frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d_k}} \tag{1}$$

$$\boldsymbol{Attention} = \text{softmax}(\boldsymbol{AttMat})\boldsymbol{V} \tag{2}$$

式中，$\boldsymbol{Q}$，$\boldsymbol{K}$，$\boldsymbol{V}$ 的形状分别为 (L_q, d_q)，(L_k, d_k)，(L_v, d_v)，$d_q = d_k = d_v$。similarity 表示计算 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$ 的相似度，softmax 表示 softmax 函数。得到的注意力矩阵 $\boldsymbol{AttMat}$ 的形状为 (L_q, L_k)。

多头注意力（multi-head attention，MHA）由多个并行的缩放点乘注意力组成，每个并行部分称为一个头，多个头的输出结果沿着通道维度拼接，最后再经过一个线性层，得到最终的输出结果。当 $\boldsymbol{Q}$，$\boldsymbol{K}$，$\boldsymbol{V}$ 由同一个输入数据经过线性变换得到时，称为多头自注意力（multi-head self-attention，MHSA）；当 $\boldsymbol{K}$ 和 $\boldsymbol{V}$ 由同一个输入数据经过变换得到、而 $\boldsymbol{Q}$ 由另一个不同输入数据变换得到时，称为多头交叉注意力（multi-head cross attention，MHCA）。

2.2　基线模型及其延迟分析

本文希望将 SwinTrack 部署在国产边缘计算主机 HS240 上。关于 HS240 及对应的计算图编程框架 Lyngor 的详细内容见附录 A。由于原始的 SwinTrack 中使用的主干网络是 SwinTransformer（Liu 等，2021），其中有部分操作目前在 Lyngor 中不支持，所以将它换成为 ConvNeXt（Liu 等，2022）。此外，由于 SwinTrack 中使用层归一化（layer normalization，LN）作为归一化层，它在 Lyngor 中支持不够好，速度较慢。本文将层归一化全部换成速度更快的批量归一化（batch normalization，BN），并做适当改进。作为基线模型，如果没做特殊声明，使用的都是 Tiny 规模。

基线模型的结构可以分为 4 个主要部分，分别是：主干网络（backbone）、编码器（encoder）、解码器（decoder）和预测头（prediction head）。解码器和预测头之间还有一个归一化层，它的延迟占比很小，将其划分到解码器中。各个组成部分的延迟如表 1 和图 2 所示。

SwinTrack 是一种孪生结构跟踪器，它的主干网络包含搜索分支（x 分支）和模板分支（z 分支）两个分支，表 1 中主干网络的延迟指的是搜索分支的延迟。表中没有列出模板分支主干网络的延迟，是因为推理阶段仅在初始化的时候执行模板分支的主干网络。总延迟指的是推理阶段的总延迟，包括搜索分支的主干网络、编码器、解码器以及预测头的延迟。

表 1 包含 4 组实验，设置和分析分别如下：

1）基线组。将 SwinTrack 中的主干网络换成 ConvNeXt，同时将层归一化换成批量归一化并进行适当改进，作为基线组。注意到，它仅使用了主干网络的前 3 个阶段的输出。由表 1 可以看到，基线组

的推理总延迟为24.2 ms，帧率41.3帧/s。它的延迟主要来自于编码器，占总延迟的56.7%。

2）A组。将基线组中的GELU（Gaussian error linear unit）（Hendrycks和Gimpel，2023）换成ReLU（rectified linear unit）（Hinton和Nair，2010），得到A组。此时总延迟降为22.0 ms，帧率略微提高，达到45.4帧/s。

3）B组。将基线组中的多头注意力全部换成LinDA结构，得到B组。关于LinDA结构详见3.1节。此时，延迟降至18.1 ms，降低6.1 ms，帧率提高到55.2帧/s。

4）C组。将A组中的多头注意力结构换成LinDA结构（相当于将B组中的GELU换成ReLU），得到C组。此时帧率达到了61.6帧/s。

由于ConvNeXt输出的数据形状是（N, C, H, W），而编码器和解码器中的数据形状是（N, L, C），因此，需要对ConvNeXt的输出进行flat和transpose操作。表1中主干网络部分的延迟是经过flat和transpose之后的。解码器的输入是指直接输入给多头注意力结构的输入。位置编码的延迟可以忽略不计，所以未在表中单独列出。在以上及后续各组实验中，位置编码部分的归一化层都是用的层归一化。预测头中仅包含线性层和激活层，所以在不同实验组中可能延迟相同。

表1　基线模型各组成部分延迟

Table 1　Latency of each component in the baseline model

部分		主干	编码器	解码器	预测头	总延迟/ms	总帧率/(帧/s)
	输入形状:	(1,3,224,224)	(1,245,384)	(1,196,384)和(1,245,384)	(1,196,384)		
	输出形状:	(1,196,384)	(1,245,384)	(1,196,384)	(1,1,14,14)和(1,14,14,4)		
基线组(GELU)		6.4 (26.4%)	13.7 (56.7%)	3.7 (15.3%)	0.4 (1.6%)	24.2 (100.0%)	41.3
A组(ReLU)		**5.2**	13.0	3.5	**0.3**	22.0	45.4
B组(GELU, LinDA)		6.4	8.6	2.7	0.4	18.1	55.2
C组(ReLU, LinDA)		**5.2**	**8.2**	**2.5**	**0.3**	**16.2**	**61.6**

注：加粗字体表示各列最优结果。

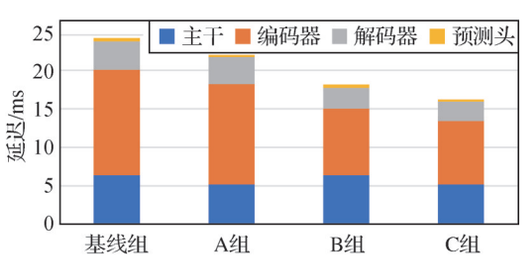


图2　不同实验组各组分延迟

Fig. 2　Latency of each component for different experimental groups

由基线组和A组对比、B组和C组对比可知，将GELU换成ReLU后，延迟分别降低了2.2 ms和1.9 ms，说明在HS240上ReLU比GELU更快。从基线组来看，此时延迟占主要部分的是编码器，所以进一步测量了基线组单层编码中令牌混合器（token mixer）和通道混合器（channel mixer）的延迟，如表2所示，发现令牌混合器的延迟占主要部分。本文将在2.3节详细介绍令牌混合器中多头注意力的延迟构成。基线组中的主干网络的延迟也占了26.4%，但是由于它是卷积结构，在底层计算的时候已经经过充分优化，所以优化空间并不大。

总的来说，将SwinTrack部署在边缘端的推理延迟瓶颈体现在以下两方面：1）GELU的硬件不友好；2）多头注意力结构。

表2　基线组中单层编码器的延迟

Table 2　Latency of single-layer encoder in the baseline group

序号	结构	延迟/ms
(a)	单层编码器	8.44
(b)	去掉令牌混合器	**0.93**(−7.51,11.0%)
(c)	去掉通道混合器	3.79(−4.65,44.9%)
(d)	将 $\boldsymbol{AttMat}$ 换成参数矩阵	4.80(−3.64,56.9%)
(e)	去掉 softmax	7.11(−1.33,84.2%)
(f)	将 $\boldsymbol{AV}$ 换成 $\boldsymbol{A}\oplus\boldsymbol{V}$	7.79(−0.65,92.3%)

注：加粗字体表示最优结果。

2.3 多头注意力结构的延迟构成

由图2中的基线组可知，除了主干网络之外，延迟还主要来自于编码器（橙色区域）和解码器（灰色区域）。由于主干网络使用的是经过充分优化的卷积架构，再次优化的空间不大，所以本文侧重于优化编码器和解码器。编码器和解码器的结构相似，均是由一个令牌混合器和通道混合器组成，其中，令牌混合器是多头自注意力或多头交叉注意力结构，通道混合器是多层感知机（multi-layer perceptron，MLP）结构，两者均具有残差连接。

以基线组中的单层编码器为例，分析其延迟的主要构成。如表2中的（a）组所示，单层编码器的延迟为 8.44 ms。在（a）组中去掉令牌混合器之后得到（b）组，延迟降低为0.93 ms，仅占原来的11.0%；若在（a）组中去掉通道混合器得到（c）组，延迟降为3.79 ms，为原来的44.9%。由此可见，延迟主要来自于令牌混合器部分。而通道混合器部分虽然也有很大计算量，但由于主要是矩阵计算，这在大多数算法库中已经被充分优化过，所以延迟很低。此外，注意到（b）组和（c）组中的延迟相加明显小于（a）组，这是正常的。实际上，在边缘端，两个操作单独使用的延迟之和与连用的延迟没有必然的大小关系。本文称之为延迟的不可加性，详见附录C。

保持通道混合器不变，修改令牌混合器中的操作，测试延迟变化情况。在令牌混合器中，输入首先经过线性层，得到 $\boldsymbol{Q}$，$\boldsymbol{K}$，$\boldsymbol{V}$，然后分别经过式（1）和式（2）得到 $\boldsymbol{AttMat}$ 和 $\boldsymbol{Attention}$，后者再经过一个线性得到输出。由于线性层的延迟很低，所以映射仅考虑其他部分。将（a）组中计算注意力矩阵 $\boldsymbol{AttMat}$ 的部分换成一个同形状的可学习参数矩阵，得到（d）组，测得延迟降为 4.80 ms。如果去掉（a）组中的softmax 操作，得到（e）组，延迟降为7.11 ms。虽然单个非线性softmax 层就占用了1.33 ms的延迟，但由于其数量少，所以优化空间并不大。

由于 $\boldsymbol{AV}$ 操作不能直接去掉，所以将它换成 $\boldsymbol{A}\oplus\boldsymbol{V}$，延迟则降为7.79 ms。这里的 $\boldsymbol{A}$ 指式（2）中的softmax（$\boldsymbol{AttMat}$），而⊕指类积和，详见附录E。虽然 $\boldsymbol{AV}$ 部分相对于注意力矩阵部分计算量也很大，但它只是简单的矩阵乘法，经过充分的底层优化后，延迟并不高。

总体而言，多头注意力结构的延迟构成总结如下：1）在多头注意力结构中，延迟主要来自于注意力矩阵的计算，其中包含了矩阵转置、矩阵乘法和元素间乘法；2）单个softmax 操作虽然延迟也不小，但是由于数量少，所以影响并不大；3）$\boldsymbol{AV}$ 的计算量虽然也较大，但由于只包含矩阵计算，这经过充分优化，所以延迟也不大。

注意到在 EfficientFormer（Li 等，2022）中，研究者认为多头自注意力结构不是速度瓶颈，该结论是基于 iPhone12 设备和 CoreML 框架上将 DeiT-Small（Touvron 等，2021）和 LeViT-256（Graham 等，2021）对比，虽然前者有更多的 MHSA、更大的计算量，但是由于具有更少的 reshape 操作，所以延迟甚至更低。本文不否认这个观点，但这不意味着多头自注意力结构的延迟不能进一步降低。图2证明提出的 LinDA 结构能有效降低延迟。

3 方 法

3.1 线性分解注意力机制

在2.3节中通过实验发现，MHA 中的延迟主要来自于注意力矩阵的计算。本节介绍一种线性分解方法，来降低其延迟和计算量。

式（1）中 $\boldsymbol{AttMat}=(attmat_{ij})$ 的每一项，其值为

$$attmat_{ij}=\frac{1}{\sqrt{d_k}}\sum_{m=1}^{d_k}q_{i,m}\times k_{j,m} \tag{3}$$

首先，给出 Transformer（Vaswani 等，2017）中的两条假设，它们在后续推导中将会用到。

假设1（独立同分布假设）：$q_{i,j}$ 和 $k_{m,n}$ 相互独立且服从相同的分布。

假设2（零期望假设）：$q_{i,j}$ 和 $k_{m,n}$ 的期望为0。

3.1.1 考虑 $q_{i,m}\times k_{j,m}$ 的期望和方差

本文的目标是希望能用单个随机变量的线性组合来近似表示两个随机变量相乘得到的随机变量。根据假设1中的相互独立性，可得

$$\begin{aligned}&E(q_{i,m}\times k_{j,m})=\\&E(k_{j,m})E(q_{i,m})+E(q_{i,m})E(k_{j,m})-E(q_{i,m})E(k_{j,m})\end{aligned} \tag{4}$$

$$\begin{aligned}&Var(q_{i,m}\times k_{j,m})=E(k_{j,m})^2Var(q_{i,m})+\\&E(q_{i,m})^2Var(k_{j,m})+Var(q_{i,m})\cdot Var(k_{j,m})\end{aligned} \tag{5}$$

式（4）显而易见，式（5）的推导过程详见附录D。

如果令 $bias_{i,j,m}$ 是期望为0、方差为 $Var(q_{i,m})\times Var(k_{i,m})$ 的随机变量，且

$$\begin{aligned}z_{i,j,m}=&E(k_{j,m})q_{i,m}+E(q_{i,m})k_{j,m}-\\&E(q_{i,m})E(k_{j,m})+bias_{i,j,m}\end{aligned} \tag{6}$$

则随机变量 $z_{i,j,m}$ 与 $q_{i,m} \times k_{j,m}$ 具有相同的期望和方差。因此，$z_{i,j,m}$ 可以作为 $q_{i,m} \times k_{j,m}$ 的一个良好的近似。另一方面，根据假设1可知，对于不同的 $q_{i,m}$ 和 $k_{j,m}$ 都是同分布的，令 $E(k_{j,m}) = a$，$E(q_{i,m}) = b$，$bias_{i,j,m} = bias$，则

$$\begin{aligned} attmat_{ij} &\approx \frac{1}{\sqrt{d_k}} \sum_{m=1}^{d_k} z_{i,j,m} = \\ &\frac{1}{\sqrt{d_k}} \sum_{m=1}^{d_k} (E(k_{j,m}) q_{i,m} + E(q_{i,m}) k_{j,m} - \\ &\quad E(q_{i,m}) E(k_{j,m}) + bias_{i,j,m}) = \\ &\frac{1}{\sqrt{d_k}} \sum_{m=1}^{d_k} (a q_{i,m} + b k_{j,m} - \\ &\quad E(q_{i,m}) E(k_{j,m}) + bias_{i,j,m}) = \\ &\frac{1}{\sqrt{d_k}} \sum_{m=1}^{d_k} (a q_{i,m} + b k_{j,m}) - \frac{1}{\sqrt{d_k}} \sum_{m=1}^{d_k} (E(q_{i,m}) E(k_{j,m})) + \\ &\frac{1}{\sqrt{d_k}} \sum_{m=1}^{d_k} bias_{i,j,m} \end{aligned} \tag{7}$$

根据假设2，$q_{i,m}$ 和 $k_{j,m}$ 的期望为0，此时 $\frac{1}{\sqrt{d_k}} \sum_{m=1}^{d_k} q_{i,m}$ 和 $\frac{1}{\sqrt{d_k}} \sum_{m=1}^{d_k} k_{j,m}$ 的期望也为0、方差分别与 $q_{i,m}$ 和 $k_{j,m}$ 的方差相等。而 $\frac{1}{\sqrt{d_k}} \sum_{m=1}^{d_k} bias_{i,j,m}$ 的期望依然为0、方差和 $bias$ 的方差相等。因此，可以得到注意力矩阵的近似表示，具体为

$$\begin{aligned} \boldsymbol{AttMat} &= (attmat_{ij})_{L_q \times L_k} \approx (a \times \boldsymbol{Q}) \oplus (b \times \boldsymbol{K}^{\mathrm{T}}) - \\ &\quad \frac{1}{\sqrt{d_k}} E(\boldsymbol{Q}) E(\boldsymbol{K})^{\mathrm{T}} + \boldsymbol{bias} \end{aligned} \tag{8}$$

式中，符号⊕表示类积和，其定义可参考附录E。虽然式(8)出现了 $\boldsymbol{K}$ 的转置，但是在代码实现上，只需要适当调整sum和unsqueeze的坐标轴即可，并不需要实际做转置操作。

将MHA和LinDA进行对比，如表3所示。LinDA和MHA的区别仅仅在于计算注意力矩阵的方式不同，其他地方都是相同的。LinDA的注意力矩阵由简单的向量点乘和矩阵内求和、矩阵间求和组成，将在4.1节看到，它比MHA具有更低的计算成本和延迟。

图3是LinDA和缩放点乘注意力结构图，图中只展示了单个头，且省略了输入和输出的投影层。

3.1.2 静态注意力矩阵的统计

在LinDA中，需要计算 $E(\boldsymbol{Q})$ 和 $E(\boldsymbol{K})$。只需要在测试阶段，在整个测试集上对 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$ 沿着批次的维度取平均即可。

3.2 知识蒸馏

实验发现，单独使用线性分解会导致精度损失，实验结果详见4.2节。本节介绍通过两种知识蒸馏来弥补这种损失。

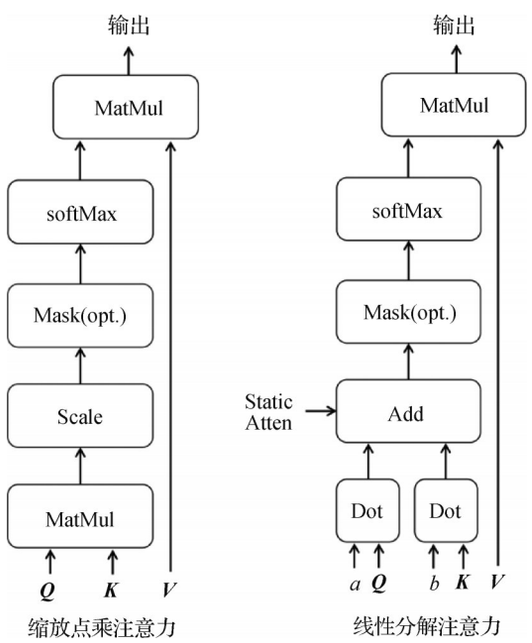


图3 缩放点乘注意力和线性分解注意力

Fig. 3 Scaled dot-product attention and linearly decomposed attention

表3 LinDA与多头注意力对比

Table 3 Comparison between LinDA and MHA

注意力	注意力矩阵	系数归一化	特征重组
MHA	$\boldsymbol{AttMat} = \frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d_k}}$	$\boldsymbol{A} = \mathrm{softmax}(\boldsymbol{AttMat})$	$\boldsymbol{Attention} = \boldsymbol{AV}$
LinDA	$\boldsymbol{AttMat} = (a \times \boldsymbol{Q}) \oplus (b \times \boldsymbol{K}^{\mathrm{T}}) - \frac{1}{\sqrt{d_k}} E(\boldsymbol{Q}) E(\boldsymbol{K})^{\mathrm{T}} + \boldsymbol{bias}$	$\boldsymbol{A} = \mathrm{softmax}(\boldsymbol{AttMat})$	$\boldsymbol{Attention} = \boldsymbol{AV}$

3.2.1　SwinTrack 中的损失函数

SwinTrack 中的损失函数包括两部分，分别是分类损失和包围框回归损失，最终损失是这两部分的加权和，具体为

$$L_{\mathrm{cls}}^{\mathrm{Sup}} = \mathrm{VFL}(p, \mathrm{IoU}(bbox, bbox^{\mathrm{gt}})) \tag{9}$$

$$L_{\mathrm{reg}}^{\mathrm{Sup}} = \sum_{j} 1_{\{q>0\}}[\, p \mathrm{L}_{\mathrm{GIoU}}(bbox_j, bbox^{\mathrm{gt}})] \tag{10}$$

$$L = w_1 L_{\mathrm{cls}}^{\mathrm{Sup}} + w_2 L_{\mathrm{reg}}^{\mathrm{Sup}} \tag{11}$$

式中，VFL 表示 varifocal loss，最早由 Zhang 等人(2021)在目标检测任务中提出，后被 Lin 等人(2022)用于跟踪任务中，其具体表达式参考 SwinTrack。$\mathrm{L}_{\mathrm{GIoU}}$ 表示广义 IoU 损失(Rezatofighi 等，2019)。p 表示预测的 IoU 感知的分类得分(IoU-aware classification score，IACS)，q 表示目标得分，$q>0$ 的区域表示正样本区域。1_{Event} 表示示性函数，Event 为事件，当事件发生时取值为 1，当事件未发生时取值为 0，此处 Event = $\{q>0\}$。$bbox$ 和 $bbox^{\mathrm{gt}}$ 分别表示模型预测的包围框和真实包围框。w_1 和 w_2 是人工设定的加权系数，在 SwinTrack 官方代码实现(Lin，2021)中，$w_1 = w_2 = 1.5$。

教师模型的作用是给学生模型提供一个新的监督信号，与真实标签不同的是，教师模型的监督信号蕴含更多的隐式知识。如何利用这些隐式知识是一个重要的问题。本文提出基于关系匹配和硬标签的知识蒸馏来充分利用教师模型的隐式知识，它们直接对学生模型的高层语义输出进行监督，隐式地让中间层特征学习合适的分布，从而做出适当的预测。

3.2.2　基于关系匹配的知识蒸馏

教师模型的隐式知识不一定是体现在它预测的精确值上，也可能隐含在各个预测结果的相对大小中。这种相对大小也称为关系(relation)(Huang 等，2022)。定义基于关系匹配的分类损失和回归损失分别为

$$L_{\mathrm{cls}}^{\mathrm{Rela}} = \frac{1}{N_{\mathrm{pos}}} \sum_{j=1}^{N_{\mathrm{Pos}}} 1_{\{q>0\}} d_{\mathrm{Pear}}(p_j^{\mathrm{stu}}, p_j^{\mathrm{tea}}) \tag{12}$$

$$L_{\mathrm{reg}}^{\mathrm{Rela}} = \frac{1}{N_{\mathrm{pos}}} \sum_{j=1}^{N_{\mathrm{Pos}}} 1_{\{q>0\}} d_{\mathrm{Pear}}(iou^{\mathrm{stu}}, iou^{\mathrm{tea}}) \tag{13}$$

式中，d_{Pear} 表示 Pearson 距离(Huang 等，2022)，p^{stu} 和 p^{tea} 分别表示学生模型和教师模型预测的 IACS，N_{pos} 表示正样本数，$iou^{\mathrm{stu}} = \mathrm{IoU}(bbox^{\mathrm{stu}}, bbox^{\mathrm{gt}})$，$iou^{\mathrm{tea}} = \mathrm{IoU}(bbox^{\mathrm{tea}}, bbox^{\mathrm{gt}})$，具体如图 4 所示。

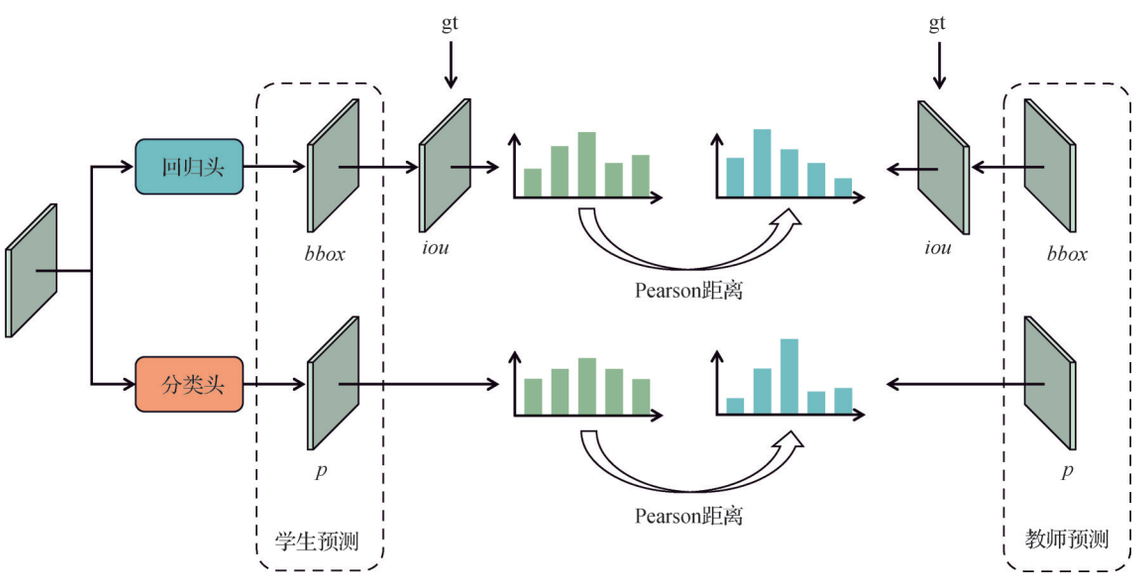


图 4　关系匹配知识蒸馏

Fig. 4　Relation matching knowledge distillation

3.2.3　基于硬标签的知识蒸馏

借鉴 Touvron 等人(2021)的方法，将教师模型预测的结果当做另一种监督目标。在 SwinTrack 中，解码器输出一个空间形状为 14 × 14 的响应图，教师模型的分类头和回归头分别基于这个响应图中每个空间位置的特征预测分类得分和包围框，分类得分最大的位置对应的包围框，被当做是教师模型最终预测的包围框 $bbox^{\mathrm{tea}}$。将式(9)和式(10)中的 $bbox^{\mathrm{gt}}$ 换成

$bbox^{\text{tea}}$，得到硬标签知识蒸馏损失，具体为

$$L_{\text{cls}}^{\text{Hard}} = \text{VFL}(p, \text{IoU}(bbox^{\text{stu}}, bbox^{\text{tea}})) \tag{14}$$

$$L_{\text{reg}}^{\text{Hard}} = \sum_{j} 1_{\{q>0\}}[p\text{L}_{\text{GIoU}}(bbox_j^{\text{stu}}, bbox_j^{\text{tea}})] \tag{15}$$

式中，$bbox^{\text{stu}}$与式(9)和式(10)中的$bbox$含义相同，只是为了强调这是学生模型，所以才加上标stu。

对于响应图中的空间位置j（$j=0,1,\cdots,14\times14-1$），$bbox_j$表示在该位置预测的包围框，$p_j$则反映了在该位置预测的包围框的精确程度。由于目标区域会有不同的干扰，如遮挡、形变及旋转等，导致在不同位置进行预测的难易程度发生变化，预测结果的精确程度也发生变化。虽然教师模型预测的结果并没有真实包围框准确，但由于前者隐式地反映了在不同位置预测的难易程度，也反映了在不同干扰下预测结果的精确程度，引入了真实标签中所缺少的一些有用信息，所以对学生模型也能起到较好的监督效果。

最终的损失为原损失函数和两种蒸馏损失的加权和，即

$$L = w_1 L_{\text{cls}}^{\text{Sup}} + w_2 L_{\text{reg}}^{\text{Sup}} + w_3 L_{\text{cls}}^{\text{Hard}} + w_4 L_{\text{reg}}^{\text{Hard}} + w_5 L_{\text{cls}}^{\text{Rela}} + w_6 L_{\text{reg}}^{\text{Rela}} \tag{16}$$

4 实　验

4.1 LinDA结构计算成本和延迟分析

4.1.1 计算成本

计算成本可以用乘加累积操作数（multiply-accumulate operations，MACs）来衡量，且只考虑MACs和长度呈二次关系的环节，即注意力矩阵部分和特征重组部分。令B表示批量大小，N_h表示头数，D_h表示每个头的维数，C_h表示隐层维数，且满足$C_h=N_hD_h$。令L_q，L_k，L_v分别表示$\boldsymbol{Q}$，$\boldsymbol{K}$，$\boldsymbol{V}$的长度。则$\boldsymbol{Q}$，$\boldsymbol{K}$，$\boldsymbol{V}$的形状分别为(B, N_h, L_q, D_h)，(B, N_h, L_k, D_h)，(B, N_h, L_v, D_h)。

对于MHA结构，注意力矩阵部分的MACs为$BN_hL_qL_kD_h$，特征重组部分的MACs为$BN_hL_qL_kD_h$，总的MACs为

$$\Omega(\text{MHA}) = 2BN_hL_qL_kD_h = 2BL_qL_kC_h \tag{17}$$

对于LinDA结构，注意力矩阵部分的MACs为$BN_hL_qD_h+BN_hL_kD_h$，特征重组部分与MHA的相同，因此总的MACs为

$$\Omega(\text{LinDA}) = BN_hL_qD_h + BN_hL_kD_h + BN_hL_qL_kD_h = B(L_q+L_k)C_h + BL_qL_kC_h \tag{18}$$

通常$L_q=L_k>100$，因此$\dfrac{BL_qL_kC_h}{B(L_q+L_k)C_h}=L_q/2>50$，所以可以近似认为

$$\Omega(\text{LinDA}) \approx BL_qL_kC_h = \frac{1}{2}\Omega(\text{MHA}) \tag{19}$$

也就是说，LinDA结构的计算成本大约只有MHA的一半，且长度越长时，前者的优势越明显。

4.1.2 延迟

如表1所示，将基线组与B组对比、A组与C组对比。基线组中的编码器和解码器部分使用的是MHSA和MHCA，其延迟为13.7 ms和3.5 ms。当将基线组的MHSA和MHCA换成LinDA之后，得到B组，对应延迟降为8.6 ms和2.5 ms。分别降低了5.1 ms和1.0 ms。对于A组和C组，对应降低了4.8 ms和1.0 ms。说明LinDA比MHA具有更低的延迟。

4.2 LinDA和知识蒸馏精度对比实验

4.2.1 LinDA与MHA对比

将基线组中编码器和解码器部分的多头注意力换成LinDA结构，静态注意力使用3.1节中描述的方法赋值。其他配置保持不变。将该模型在跟踪数据集上训练并测试，实验结果如表4所示，在LaSOT（Fan等，2019）验证集上的SUC指标为63.9%，在LaSOT_ext（Fan等，2021）验证集上的SUC指标为42.1%，比基线组分别降低了3.1%和3.9%。说明在这种训练方式下，LinDA不如多头注意力结构。

4.2.2 使用教师模型预测头参数初始化

Chen等人（2022b）认为，复用教师头能显著弥补教师模型和学生模型之间的性能差距，本文也将教师模型中的预测头用于LinDA中。不同于Chen等人（2022b），本文在训练阶段，依然对头部进行学习。使用教师模型的预测头参数初始化后，在LaSOT和LaSOT_ext上的SUC分别达到64.6%和43.5%。说明教师模型预测头确实有助于弥补性能差距。

4.2.3 关系匹配和硬标签知识蒸馏

虽然使用教师模型的预测头参数初始化能在一定程度上弥补精度差距，但此时依然不如基线组。本文在此基础上增加关系匹配知识蒸馏，在LaSOT/LaSOT_ext上提高到65.0%/44.5%；再增加硬标签

表 4　精度消融实验

Table 4　Ablation experiments for precision

/%

实验	描述	LaSOT			LaSOT_ext		
		SUC	PRE	NPRE	SUC	PRE	NPRE
SwinTrack-T	原始的	68.1	72.2	77.4	47.4	53.9	54.5
基线组	主干网络: ConvNeXt	67.0	**70.8**	75.9	**46.0**	**52.0**	**52.9**
LinDA 和蒸馏	LinDA	63.9	67.3	72.7	42.1	46.7	48.1
	+教师预测头初始化	64.6	67.9	73.4	43.5	49.1	49.8
	+关系匹配知识蒸馏	65.0	68.2	73.9	44.5	50.2	51.2
	+硬标签知识蒸馏	66.0	69.3	74.8	45.1	50.9	51.6
	+ early droppath	**67.1**	70.3	**76.2**	44.6	50.5	51.1
LinDATrack	LinDA，蒸馏	66.3	69.8	75.7	45.8	51.8	52.8

注：加粗字体表示各列除 SwinTrack-T 之外的最优结果。PRE：precisionlNPRE；normed precision。

知识蒸馏，分别提高到 66.0%/45.1%。最后，还使用了一个小技巧，训练阶段在前 120 个 epochs 中，droppath rate 线性衰减到 0，此时在 LaSOT 上的 SUC 进一步提高到 67.1%，比基线组高 0.1%；而在 LaSOT_ext 上有所下降，降为 44.6%。

LinDATrack 同时使用 GELU、LinDA 和知识蒸馏，得到最终的 LinDATrack，它在 LaSOT 和 LaSOT_ext 上的 SUC 指标分别为 66.3% 和 45.8%，相对基线组仅仅降低 0.7% 和 0.2%，这对于边缘端是可接受的。

本文还探究了 LinDA 结构中静态注意力和线性分解系数起的作用，实验结果如表 5 所示。

表 5　LinDA 对比实验和消融实验

Table 5　Comparison and ablation experiments for LinDA

/%

项目	描述	LaSOT			LaSOT_ext		
		SUC	PRE	NPRE	SUC	PRE	NPRE
LinDA	使用蒸馏，默认静态注意力，默认线性分解系数	**67.1**	**70.5**	**76.0**	44.7	50.2	51.0
静态注意力对比	不使用蒸馏，去掉静态注意力	62.5	64.9	70.8	42.2	46.9	48.2
	不使用蒸馏，默认静态注意力	63.9	67.3	72.7	42.1	46.7	48.1
	使用蒸馏，随机静态注意力	66.7	70.0	75.6	**45.7**	**51.7**	**52.2**
线性分解系数对比	不使用蒸馏，去掉系数	63.0	65.7	71.5	40.1	43.7	45.6
	使用蒸馏，去掉系数	66.6	70.0	75.4	44.7	50.7	51.1
前馈注意力	不使用蒸馏	61.2	63.5	69.3	40.2	44.3	45.5

注：加粗字体表示各列最优结果。

4.2.4　静态注意力矩阵的取值

在 3.1 节中，得到的静态注意力矩阵需要使用预先统计的值。本文设置实验探究不使用统计值的情况。首先不使用知识蒸馏，同时去掉静态注意力，得到在 LaSOT 和 LaSOT_ext 上的 SUC 指标分别为 62.5% 和 42.2%；然后加上默认静态注意力，也就是使用统计值，得到在 LaSOT 和 LaSOT_ext 上的 SUC 指标分别为 63.9% 和 42.1%，在 LaSOT 上增加了 1.4%，在 LaSOT_ext 上仅降低了 0.1%。由此可见，在不使用蒸馏的情况下，统计信息确实起到了较大作用。本文还考虑了使用知识蒸馏和随机静态注意力矩阵的情况，对静态注意力矩阵使用截断正态分

布赋值，得到LaSOT和LaSOT_ext上的SUC指标分别为66.7%和45.7%，分别降低0.4%和增加1.0%。本文认为这种在一个数据集上的指标略微增加、在另一个数据集略微降低的现象由随机因素导致，总体来说可以认为近似不变。这表明，在使用知识蒸馏的情况下，静态注意力矩阵的取值没那么重要。这与在3.1节中的推导并不矛盾，在式(8)中，使用的***bias***矩阵和静态注意力矩阵是同型的，静态注意力矩阵中的信息完全有可能被吸收进***bias***矩阵中。事实上，在实验中发现，加上静态注意力矩阵之后，模型的收敛速度更快。这表明静态注意力矩阵引入了一些先验信息，这些信息能通过可学习参数直接学到。

4.2.5　线性分解系数

式(8)中有两个系数a和b，它们是可学习参数。默认设置下，令它们的维度和MHA中每个预测头的维度相等。去掉这两个系数的结果如表5所示，在不使用知识蒸馏的情况下去掉系数，得到在LaSOT和LaSOT_ext上的SUC指标为63.0%和40.1%，相对于使用系数降低了0.9%和2.0%；在使用知识蒸馏的情况下去掉系数，指标分别为66.6%和44.7%，在LaSOT数据集上降低了0.5%，而在LaSOT_ext数据集上不变。这两种情况下，静态注意力矩阵均使用统计值。这表明，无论是否使用知识蒸馏，线性分解系数对LinDA结构都是有效的。

4.2.6　LinDA与前馈注意力对比

LinDA结构是一种加法注意力，而在Transformer（Vaswani等，2017）中提到，注意力矩阵除了可以使用点积来计算之外，还可以使用加法来计算。这种加法注意力最早由Bahdanau等人(2016)提出，用在双向循环神经网络(bidirectional recurrent neural network，BRNN)结构中，由一个具有两层的前馈网络(feedforward network，FFN)实现，即在两个线性层中间加一个激活层。本文对其做适当修改，应用在多头注意力结构中。由于$\boldsymbol{Q}$和$\boldsymbol{K}$已经经过了一层线性层，同时也是为了尽可能保持参数量一致，所以省略掉第一个线性层。另一方面，在Bahdanau等人(2016)方法中使用的激活函数是tanh函数，由于其容易造成梯度消失，故将它换成常用的GELU函数。由表5可知，在不使用蒸馏的情况下，基于FFN的加法注意力在LaSOT/LaSOT_ext上的SUC分别为61.2%/40.2%，而LinDA则是63.9%/42.1%，FNN比LinDA分别降低了2.7%和1.9%，这说明LinDA结构更有效。

4.2.7　速度和功耗对比

本文使用LaSOT(Fan等，2019)数据集测试，比较了LinDATrack在安擎EG84G-G20服务器和灵汐HS240边缘计算主机上推理阶段的速度和功耗，如表6所示。HS240上使用的驱动和工具包版本分别是LynDriver1.10.0和LynSDK1.10.0。安擎EG84G-G20搭载英伟达GeForce RTX 3090显卡，其算力为285TFLOPS@FP16和35.6TFLOPS@FP32，内存24 GB，带宽936 GB/s，推理速度91.8帧/s，推理阶段整机功率为1 273.0 W，单独的RTX 3090功率是126.5 W。灵汐边缘计算主机HS240搭载的是灵汐自研的KA200类脑芯片，算力为32TOPS@INT8和16TFLOPS@FP16，不支持FP32及更高精度运算。本文的算法部署时采用的是FP16精度，在此精度下，KA200的算力仅仅是RTX3090的5.6%，内存8 GB，带宽25.6 GB/s，推理速度61.6帧/s，推理阶段整机功率为79.5 W，单独的KA200芯片功率是5.0 W。边缘端HS240上的整机功率仅占服务器端的6.2%，功耗下降高达93.8%。在多个硬件参数远低于RTX 3090的情况下，KA200依然具有较高的推理速度，充分说明了LinDATrack的边缘端性能优势。

4.2.8　参数量和FLOPs对比

本文对比了LinDATrack与其他算法的参数量和浮点运算量(floating point operations，FLOPs)，如

表6　不同设备上的速度和功耗对比

Table 6　Comparison of speed and power consumption on different devices

设备	算力	制程工艺/nm	内存/GB	带宽/(GB/s)	速度/(帧/s)	整机功率/W	芯片功率/W
EG84G-G20(RTX3090)	285TFLOPS@FP16; 35.6TFLOPS@FP32	8	24	936	91.8	1 273.0	126.5
HS240(KA200)	32TOPS@INT8; 16TFLOPS@FP16	12	8	25.6	61.6	79.5	5

表 7 所示。SwinTrack-T 的计算量是 6.4 GB MACs（Lin 等，2022），将它换算成 12.8 G FLOPs。LinDATrack 相对 SwinTrack-T 的参数量增加了 5.1 M，主要是增加在可学习的系数上；而计算量减少了 6.5 G，一是因为将复杂的 MHA 分解成了简单的 LinDA 结构，二是将主干网络由 SwinTransformer 换成了 ConvNeXt。在主干网络中，仅使用前 3 个阶段的输出。相对于 STARK-S50（Yan 等，2021a），LinDATrack 的参数量虽然增加了 4.5 M，但是计算量反而减少了 4.2 G。相对于 SimTrack-B/32（Chen 等，2022a），计算量减少了 5.2 G。总的来说，LinDATrack 在参数量和计算代价方面具有较好的平衡，体现出较高的计算效率。

表 7　参数量和 FLOPs 对比

Table 7　Comparison of number of parameters and FLOPs

算法	参数量/M	FLOPs/G
SwinTrack-T	22.7	12.8
STARK-S50	23.3	10.5
SimTrack-B/32	–	11.5
LinDATrack	27.8	6.3

4.3　SOTA 比较

由于此前没有工作在 HS240 上部署过，所以很难公平比较其性能。但是为了看清与主流跟踪算法性能之间的大致关系，将 LinDATrack 的测试结果与服务器端和移动端最先进（state-of-the-art，SOTA）跟踪算法进行比较，结果如表 8 所示。

表 8　SOTA 比较

Table 8　SOTA comparison

设备	算法	LaSOT			LaSOT_ext			TrackingNet			GOT-10k		
		SUC	PRE	NPRE	SUC	PRE	NPRE	SUC	PRE	NPRE	AO	$SR_{0.5}$	$SR_{0.75}$
服务器端	SwinTrack-T	**68.1**	**72.2**	**77.4**	**47.4**	**53.9**	**54.5**	80.5	77.9	85.5	**69.0**	**78.1**	**62.1**
	TransT	64.9	69.0	73.8	–	–	–	**81.4**	80.3	**86.7**	67.1	76.8	60.9
	STARK-S50	65.8	–	–	–	–	–	80.3	–	85.1	67.2	76.1	61.2
	SimTrack-B/32	66.2	–	76.1	–	–	–	79.1	**83.9**	–	–	–	–
移动端	LightTrack-Mobile	53.8	53.7	–	–	–	–	72.5	69.5	77.9	61.1	71.0	–
	LightTrack-LargeA	55.0	55.2	–	–	–	–	73.6	70.0	78.8	61.5	72.3	–
	LightTrack-LargeB	55.5	56.1	–	–	–	–	73.3	70.8	78.9	62.3	72.6	–
边缘端	LinDATrack（本文）	66.3	69.8	75.7	45.8	51.8	52.8	80.0	77.4	85.0	67.1	76.2	60.9

注：加粗字体表示各列最优结果。“–”表示缺少此组数据。

4.3.1　在 LaSOT 数据集上的比较

LaSOT（Fan 等，2019）是一个大规模、长期目标跟踪数据集，具有高质量的密集标注包围框，包含 70 个类别，每个类别 20 个序列，共 1 400 个序列，其中 280 个是测试序列。从表 8 可以看到，LinDATrack 的 SUC、PRE（precision）和 NPRE（normed precision）分别为 66.3%、69.8% 和 75.7%，相较于基线算法 SwinTrack-T，SUC 仅下降 1.8%，优于服务器端其他基于 Transformer 的跟踪器，包括 TransT（64.9%）和 STARK-S50（65.8%），与单流跟踪器 SimTrack-B/32（66.2%）相当。LinDATrack 大大优于移动端算法 LightTrack（Yan 等，2021b）的各种变体，相较最佳变体 LargeB，SUC 和 PRE 分别提高 10.8% 和 13.7%。

4.3.2　在 LaSOT_ext 数据集上的比较

LaSOT_ext（Fan 等，2021）是 LaSOT 数据集的扩充，加入了额外的 150 个序列仅用于测试。这些序列来自于 15 个新类别，具有更多的小目标和干扰物，因此更具挑战性。如表 8 所示，在 LaSOT_ext 上，LinDATrack 的 SUC、PRE 和 NPRE 分别为 45.8%、51.8% 和 52.8%，相对于基线算法 SwinTrack-T，SUC 仅下降 1.6%。

4.3.3　在 TrackingNet 数据集上的比较

TrackingNet（Müller 等，2018）数据集包含 511 个测试序列，覆盖 27 种目标类别。它的测试集的标注

没有公开，需要将跟踪结果提交到在线服务器进行评测。如表8所示，在TrackingNet上，LinDATrack的SUC、PRE和NPRE分别为80.0%、77.4%和85.0%，相对于SwinTrack-T，SUC仅下降0.5%，较SimTrack-B/32提高0.9%，较移动端LinghtTrack-LargeA大幅提高了6.4%。

4.3.4 在GOT-10k数据集上的比较

GOT-10k（Huang等，2021）具有180个测试序列，它的测试集和训练集的物体类别没有重合，要求仅使用自身的训练集进行训练，用于测试算法的泛化能力。本文遵循它的协议，将跟踪结果提交到在线服务器进行测试，AO（average overlap）为67.1%，与服务器端的TranT（67.1%）和STARK-S50（67.2%）相当，远高于移动端LightTrack-LargeB（62.3%）。

5 结 论

本文提出了一种面向国产边缘计算主机HS240的目标跟踪算法。首先，详细分析了基线算法在计算主机上的延迟，发现延迟瓶颈在多头注意力结构上。然后，设计了一种面向边缘端的线性分解注意力结构，可有效降低多头注意力的计算复杂度和延迟。为了弥补精度损失，提出一种知识蒸馏方法，相对于基线算法的精度指标仅下降1%～2%。将提出的算法部署在HS240上，可达到61.6帧/s的推理速度，功耗约79.5 W，相对于服务器端，功耗下降93.8%。实验结果表明，所提算法在边缘端表现出色，具有较快的推理速度，在可接受的精度损失下，显著降低了功耗。

参考文献（References）

Bahdanau D, Cho K and Bengio Y. 2016. Neural machine translation by jointly learning to align and translate [EB/OL]. [2024-04-07]. https://arxiv.org/pdf/1409.0473.pdf

Bertinetto L, Valmadre J, Henriques J F, Vedaldi A and Torr P H S. 2016. Fully-convolutional siamese networks for object tracking//Proceedings of 2016 European Conference on Computer Vision (ECCV) Workshops. Amsterdam, the Netherlands: Springer: 850-865 [DOI: https://doi.org/10.1007/978-3-319-48881-3_56]

Bhat G, Danelljan M, van Gool L and Timofte R. 2019. Learning discriminative model prediction for tracking//Proceedings of 2019 IEEE/CVF International Conference on Computer Vision (ICCV). Seoul, Korea (South): IEEE: 6181-6190 [DOI: 10.1109/ICCV.2019.00628]

Cai H, Li J Y, Hu M Y, Gan C and Han S. 2023. EfficientViT: lightweight multi-scale attention for high-resolution dense prediction//Proceedings of 2023 IEEE/CVF International Conference on Computer Vision (ICCV). Paris, France: IEEE: 17256-17267 [DOI: 10.1109/ICCV51070.2023.01587]

Chen B Y, Li P X, Bai L, Qiao L, Shen Q H, Li B, Gan W H, Wu W and Ouyang W L. 2022a. Backbone is all your need: a simplified architecture for visual object tracking//Proceedings of the 17th European Conference on Computer Vision (ECCV). Tel Aviv, Israel: Springer: 375-392 [DOI: 10.1007/978-3-031-20047-2_22]

Chen D F, Mei J P, Zhang H L, Wang C, Feng Y and Chen C. 2022b. Knowledge distillation with the reused teacher classifier//Proceedings of 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). New Orleans, USA: IEEE: 11923-11932 [DOI: 10.1109/CVPR52688.2022.01163]

Chen X, Yan B, Zhu J W, Wang D, Yang X Y and Lu H C. 2021. Transformer tracking//Proceedings of 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Nashville, USA: IEEE: 8122-8131 [DOI: 10.1109/CVPR46437.2021.00803]

Chen Z L and Shi F H. 2022. Double template fusion based siamese network for robust visual object tracking. Journal of Image and Graphics, 27(4): 1191-1203(陈志良, 石繁槐. 2022. 结合双模板融合与孪生网络的鲁棒视觉目标跟踪. 中国图象图形学报, 27(4): 1191-1203) [DOI: 10.11834/jig.200660]

Child R, Gray S, Radford A and Sutskever I. 2019. Generating long sequences with sparse transformers [EB/OL]. [2024-04-07]. https://arxiv.org/pdf/1904.10509.pdf

Choromanski K, Likhosherstov V, Dohan D, Song X Y, Gane A, Sarlos T, Hawkins P, Davis J, Mohiuddin A, Kaiser L, Belanger D, Colwell L and Weller A. 2022. Rethinking attention with performers [EB/OL]. [2024-04-07]. https://arxiv.org/pdf/2009.14794.pdf

Cui Y T, Jiang C, Wang L M and Wu G S. 2022. MixFormer: end-to-end tracking with iterative mixed attention//Proceedings of 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). New Orleans, USA: IEEE: 13598-13608 [DOI: 10.1109/CVPR52688.2022.01324]

Danelljan M, Bhat G, Khan F S and Felsberg M. 2019. ATOM: accurate tracking by overlap maximization//Proceedings of 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Long Beach, USA: IEEE: 4655-4664 [DOI: 10.1109/CVPR.2019.00479]

Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X H, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S,

Uszkoreit J and Houlsby N. 2021. An image is worth 16x16 words: transformers for image recognition at scale [EB/OL]. [2024-04-07]. https://arxiv.org/pdf/2010.11929.pdf

Fan H, Bai H X, Lin L T, Yang F, Chu P, Deng G, Yu S J, Harshit, Mingzhen M Z, Liu J H, Xu Y, Liao C Y, Yuan L and Ling H B. 2021. LaSOT: a high-quality large-scale single object tracking benchmark. International Journal of Computer Vision, 129(2): 439-461 [DOI: 10.1007/s11263-020-01387-y]

Fan H, Lin L T, Yang F, Chu P, Deng G, Yu S J, Bai H X, Xu Y, Liao C Y and Ling H B. 2019. LaSOT: a high-quality benchmark for large-scale single object tracking//Proceedings of 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Long Beach, USA: IEEE: 5369-5378 [DOI: 10.1109/CVPR.2019.00552]

Graham B, El-Nouby A, Touvron H, Stock P, Joulin A, Jégou H and Douze M. 2021. LeViT: a vision transformer in ConvNet's clothing for faster inference//Proceedings of 2021 IEEE/CVF International Conference on Computer Vision (ICCV). Montreal, Canada: IEEE: 12239-12249 [DOI: 10.1109/ICCV48922.2021.01204]

Hao Z W, Guo J Y, Jia D, Han K, Tang Y H, Zhang C, Hu H and Wang Y H. 2022. Learning efficient vision transformers via fine-grained manifold distillation//Proceedings of the 36th International Conference on Neural Information Processing Systems. New Orleans, USA: Curran Associates Inc.: 9164-9175

Hendrycks D and Gimpel K. 2023. Gaussian error linear units (GELUs) [EB/OL]. [2024-04-07]. https://arxiv.org/pdf/1606.08415.pdf

Hinton G E and Nair V. 2010. Rectified linear units improve restricted boltzmann machines//Proceedings of the 27th International Conference on International Conference on Machine Learning. Haifa, Israel: PMLR: 807-814

Hinton G E, Vinyals O and Dean J. 2015. Distilling the knowledge in a neural network [EB/OL]. [2024-04-07]. https://arxiv.org/pdf/1503.02531.pdf

Huang L H, Zhao X and Huang K Q. 2021. Got-10k: a large high-diversity benchmark for generic object tracking in the wild. IEEE Transactions on Pattern Analysis and Machine Intelligence, 43(5): 1562-1577 [DOI: 10.1109/TPAMI.2019.2957464]

Huang T, You S, Wang F, Qian C and Xu C. 2022. Knowledge distillation from a stronger teacher//Proceedings of the 36th International Conference on Neural Information Processing Systems. New Orleans, USA: Curran Associates Inc.: 33716-33727

Kasai J, Peng H, Zhang Y Z, Yogatama D, Ilharco G, Pappas N, Mao Y, Chen W Z and Smith N A. 2021. Finetuning pretrained transformers into RNNs [EB/OL]. [2024-04-07]. https://arxiv.org/pdf/2103.13076.pdf

Li B, Yan J J, Wu W, Zhu Z and Hu X L. 2018. High performance visual tracking with siamese region proposal network//Proceedings of 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA: IEEE: 8971-8980 [DOI: 10.1109/CVPR.2018.00935]

Li Y Y, Yuan G, Wen Y, Hu J, Evangelidis G, Tulyakov S, Wang Y Z and Ren J. 2022. EfficientFormer: vision transformers at MobileNet speed//Proceedings of the 36th International Conference on Neural Information Processing Systems. New Orleans, USA: Curran Associates Inc.: 12934-12949

Lin L T. 2022. Github reposity: swintrack [CP/OL]. [2024-04-07]. https://github.com/LitingLin/SwinTrack

Lin L T, Fan H, Xu Y and Ling H B. 2021. SwinTrack: a simple and strong baseline for transformer tracking [EB/OL]. [2024-04-07]. https://arxiv.org/pdf/2112.00995.pdf

Liu Z, Lin Y T, Cao Y, Hu H, Wei Y X, Zhang Z, Lin S and Guo B N. 2021. Swin transformer: hierarchical vision transformer using shifted windows//Proceedings of 2021 IEEE/CVF International Conference on Computer Vision (ICCV). Montreal, Canada: IEEE: 9992-10002 [DOI: 10.1109/ICCV48922.2021.00986]

Liu Z, Mao H Z, Wu C Y, Feichtenhofer C, Darrell T and Xie S N. 2022. A ConvNet for the 2020s//Proceedings of 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). New Orleans, USA: IEEE: 11966-11976 [DOI: 10.1109/CVPR52688.2022.01167]

Lu J C, Yao J H, Zhang J G, Zhu X T, Xu H, Gao W G, Xu C J, Xiang T and Zhang L. 2021. SOFT: softmax-free transformer with linear complexity//Proceedings of the 35th International Conference on Neural Information Processing Systems. Virtual, Online: Curran Associates Inc.: 21297-21309

Müller M, Bibi A, Giancola S, Al-Subaihi S and Ghanem B. 2018. TrackingNet: a large-scale dataset and benchmark for object tracking in the wild//Proceedings of the 15th European Conference on Computer Vision (ECCV). Munich, Germany: Springer: 310-327 [DOI: 10.1007/978-3-030-01246-5_19]

Rezatofighi H, Tsoi N, Gwak J Y, Sadeghian A, Reid I and Savarese S. 2019. Generalized intersection over union: a metric and a loss for bounding box regression//Proceedings of 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Long Beach, USA: IEEE: 658-666 [DOI: 10.1109/CVPR.2019.00075]

Shang X R, Wen Y L, Xi X F and Hu F Y. 2021. Target tracking system based on the Siamese guided anchor region proposal network. Journal of Image and Graphics, 26(2): 415-424(尚欣茹, 温尧乐, 奚雪峰, 胡伏原. 2021. 孪生导向锚框RPN网络实时目标跟踪. 中国图象图形学报, 26(2): 415-424) [DOI: 10.11834/jig.190658]

Touvron H, Cord M, Douze M, Massa F, Sablayrolles A and Jégou H. 2021. Training data-efficient image transformers and distillation through attention//Proceedings of the 38th International Conference on Machine Learning. Virtual Event: PMLR: 10347-10357

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser Ł and Polosukhin I. 2017. Attention is all you need//Proceedings of the 31st International Conference on Neural Information Processing Systems. Long Beach, USA: Curran Associates Inc.: 6000-

6010

Wang F S, Yin S S, He B and Sun F M. 2023. A Gaussian mask-based correlation filter tracking algorithm. Journal of Image and Graphics, 28(10): 3092-3106(王法胜, 尹双双, 贺冰, 孙福明. 2023. 引入高斯掩膜的相关滤波目标跟踪算法. 中国图象图形学报, 28(10): 3092-3106)[DOI: 10.11834/jig.220856]

Wang N, Zhou W G, Wang J and Li H Q. 2021. Transformer meets tracker: exploiting temporal context for robust visual tracking//Proceedings of 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Nashville, USA: IEEE: 1571-1580 [DOI: 10.1109/CVPR46437.2021.00162]

Wu K, Zhang J N, Peng H W, Liu M C, Xiao B, Fu J L and Yuan L. 2022. TinyViT: fast pretraining distillation for small vision transformers//Proceedings of the 17th European Conference on Computer Vision (ECCV). Tel Aviv, Israel: Springer: 68-85 [DOI: 10.1007/978-3-031-19803-8_5]

Xue W L, Zhang Z B, Pei S L, Zhang K H and Chen S Y. 2024. Mixing tokens from target and search regions for visual object tracking. Journal of Computer Research and Development, 61(2): 460-469 (薛万利, 张智彬, 裴生雷, 张开华, 陈胜勇. 2024. 混合目标与搜索区域令牌的视觉目标跟踪. 计算机研究与发展, 61(2): 460-469) [DOI: 10.7544/issn1000-1239.202220698]

Yan B, Peng H W, Fu J L, Wang D and Lu H C. 2021a. Learning spatio-temporal transformer for visual tracking//Proceedings of 2021 IEEE/CVF International Conference on Computer Vision (ICCV). Montreal, Canada: IEEE: 10428-10437 [DOI: 10.1109/ICCV48922.2021.01028]

Yan B, Peng H W, Wu K, Wang D, Fu J L and Lu H C. 2021b. LightTrack: finding lightweight neural networks for object tracking via one-shot architecture search//Proceedings of 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Nashville, USA: IEEE: 15175-15184 [DOI: 10.1109/CVPR46437.2021.01493]

Ye B T, Chang H, Ma B P, Shan S G and Chen X L. 2022. Joint feature learning and relation modeling for tracking: a one-stream framework//Proceedings of the 17th European Conference on Computer Vision (ECCV). Tel Aviv, Israel: Springer: 341-357 [DOI: 10.1007/978-3-031-20047-2_20]

Zhao G X, Lin J Y, Zhang Z Y, Ren X C, Su Q and Sun X. 2019. Explicit sparse transformer: concentrated attention through explicit selection [EB/OL]. [2024-04-07]. https://arxiv.org/pdf/1912.11637.pdf

Zhang H Y, Wang Y, Dayoub F and Sünderhauf N. 2021. VarifocalNet: an IoU-aware dense object detector//Proceedings of 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Nashville, USA: IEE E: 8510-8519 [DOI: 10.1109/CVPR46437.2021.00841]

Zhu Z, Wang Q, Li B, Wu W, Yan J J and Hu W M. 2018. Distractor-aware siamese networks for visual object tracking//Proceedings of the 15th European Conference on Computer Vision (ECCV). Munich, Germany: Springer: 103-119 [DOI: 10.1007/978-3-030-01240-3_7]

作者简介

邱森波,男,博士研究生,主要研究方向为目标跟踪、知识蒸馏和模型轻量化。E-mail: miaobo.qiu@mais.ia.ac.cn

高晋,通信作者,男,研究员,主要研究方向为计算机视觉、机器学习、自主机器人和自动驾驶。E-mail: jin.gao@nlpr.ia.ac.cn

林述波,男,硕士,主要研究方向为自监督学习和多目标跟踪。E-mail: linshubo2023@ia.ac.cn

李椋,男,助理研究员,主要研究方向为类脑计算与视觉感知。E-mail: liang.li.brain@aliyun.com

王刚,男,副研究员,主要研究方向为深度学习与类脑视觉计算。E-mail: g_wang@foxmail.com

胡卫明,男,研究员,主要研究方向为计算机视觉、智能视觉监控、网络信息安全分析与识别。E-mail: wmhu@nlpr.ia.ac.cn

王以政,男,中国科学院院士,研究员,主要研究方向为神经生物学、视觉与类脑计算。E-mail: yzwang57@sina.com

附录A　国产边缘计算主机HS240简介

1)领启KA200类脑芯片。KA200是灵汐科技自研的一款类脑芯片,它基于全新的存算一体、众核并行、异构融合架构,能高效支持深度学习神经网络、生物神经网络和大规模脑仿真。单芯片集成25万神经元和2500万突触(稠密模式),可扩展支持200万神经元和20亿突触的集成计算(稀疏模式),支持混合精度计算。

2)Lyngor。Lyngor是基于KA200类脑芯片的计算图编程框架,它能对多种深度学习框架提供的模型文件进行解析和编译,生成KA200所需的指令文件作为计算图推理引擎,在KA200上加载该引擎即可进行计算图加速计算。支持的深度学习框架包括TensorFlow、PyTorch、Keras、Caffe、MXNet和ONNX等。

3)类脑计算模组HM100。HM100模组是一款基于领启KA200类脑计算芯片开发的高度集成化模组,提供强大的算力、大容量内存、最小系统以及丰富的接口,方便用户对KA200进行快速产品开

发，缩短开发周期。HM100 整板最大功耗为 15 W，搭载 1 颗 KA200 芯片。

4）边缘计算主机 HS240。HS240 是一款面向边缘应用场景的轻量边缘设备，具有完全密闭结构，防水防尘，支持快速拔插，满足在恶劣环境下的高性能图像采集、处理和使用要求。它包含 4 个 HM100 模组，额定电压直流 12 V，最大功耗 100 W，计算能力为 32TOPS*4@INT8 和 16TFLOPS*4@FP16。

附录B　编码器、解码器结构图

编码器、解码器结构如图 B1 所示。图中的编码器有 4 层多头注意力结构，解码器有 1 层多头注意力结构。搜索分支、模板分支的特征经过拉伸、拼接，分别输入编码器、解码器，得到输出特征。输出特征经过合适的 reshape，得到最终响应图。

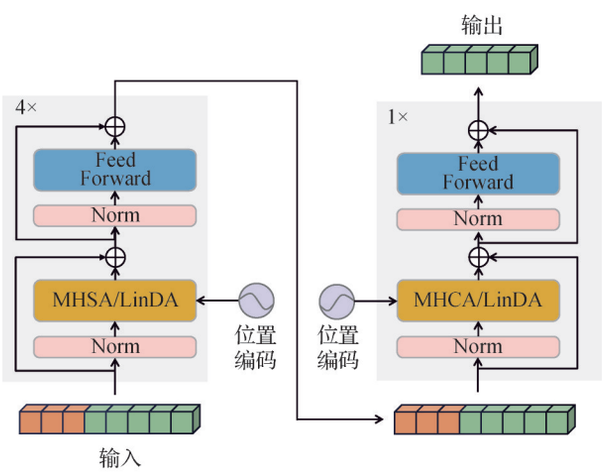


图 B1　编码器、解码器结构

Fig. B1　Structure of encoder and decoder

附录C　边缘端芯片延迟的不可加性

用 $\tau(op(x))$ 表示输入 x 经过操作 op 的延迟。延迟是与输入有关的，但是为了简便，省略输入，记做 $\tau(op)$。用 $op_2(op_1(x))$ 表示操作 op_1 和 op_2 的复合或连用，也就是输入 x 先经过 op_1，再将结果输入给 op_2（这里的复合或连用指两个操作一起编译并执行的情况，而不是单独编译），其延迟记为 $\tau(op_2(op_1(x)))$，简记为 $\tau(op_2 \circ op_1)$。

定义 1　延迟的不可加性

对于相同输入，两个操作连用的延迟与单独使用的延迟之和没有必然的大小关系，即

$$\tau(op_2 \circ op_1) < \tau(op_1) + \tau(op_2) \quad (C1)$$

$$\tau(op_2 \circ op_1) = \tau(op_1) + \tau(op_2) \quad (C2)$$

$$\tau(op_2 \circ op_1) > \tau(op_1) + \tau(op_2) \quad (C3)$$

这三者都可能发生，称这种现象为延迟的不可加性。式（C1）称为延迟的三角不等式，式（C2）称为延迟的线性，式（C3）称为延迟的上凸性。也可以仅将三角不等式和上凸性统称为不可加性或非线性。

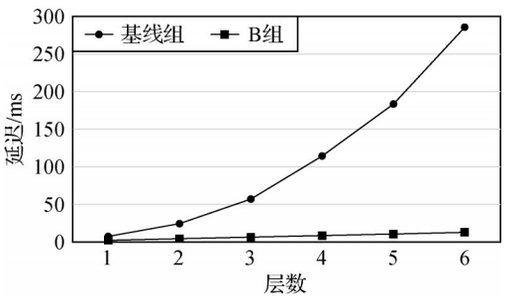
实验中在 KA200 芯片上同时观察到了这 3 种现象。

1）延迟的三角不等式。这种情况下，两个操作连用的延迟小于每个操作单独使用的延迟的和。一个典型的例子就是 Conv 层和 BN 层的连用。当两者连用之后，可以对它们进行算子融合，融合后的延迟小于单独使用的延迟之和。

2）延迟的上凸性。上凸性指的是两个操作连用的延迟大于每个操作单独使用时的延迟之和。出现这种现象通常与芯片的缓存大小或数据读取的底层实现方式有关：当代码运行所需的内存在适当的范围内或者数据读取的底层实现比较高效时，数据能提前从主存转移到缓存中，处理器需要数据的时候可以直接从缓存读取，此时延迟随层数线性增加；当所需内存超过这一范围或者数据读取的底层实现比较低效，导致部分数据无法及时读进缓存，而只能从读写速率更慢的主存读取，此时，延迟就随着层数的增加而急剧增大。

以基线组和 B 组为例来说明延迟的上凸性。表 1 中的 encoder 默认包含 4 层，每层包含一个 token mixer（MHSA 模块）和一个 channel mixer（FFN 模块）。改变基线组和 B 组的 encoder 的层数再测延迟，得到如图 C1 所示曲线。两组的延迟都是随层数的增加而增加，区别在于，在基线组中，延迟随层数增加得越来越快，而在 B 组中，延迟随层数近似线性增加。

3）延迟的线性。这种情况下，两个操作连用的延迟等于每个操作单独使用的延迟的和。图 C1 中的 B 组即表现出线性。这种现象通常出现在多个操作无法联合优化且数据读取量不大时。



图C1　延迟和层数的关系

Fig. C1　Relationship between latency and number of layers

附录D　公式推导

性质1　独立随机变量乘积的方差

已知两个随机变量x_1和x_2相互独立，则x_1与x_2相乘得到的随机变量的方差为

$$Var(x_1x_2)=Var(x_1)Var(x_2)+E(x_2)^2Var(x_1)+E(x_1)^2Var(x_2) \tag{D1}$$

证明：

$$\begin{aligned}Var(x_1x_2)&=E(x_1^2x_2^2)-E(x_1x_2)^2=\\&E(x_1^2)E(x_2^2)-E(x_1)^2E(x_2)^2=(Var(x_1)+\\&E(x_1)^2)(Var(x_2)+E(x_2)^2)-E(x_1)^2E(x_2)^2=\\&Var(x_1)Var(x_2)+Var(x_1)E(x_2)^2+\\&Var(x_2)E(x_1)^2=Var(x_1)Var(x_2)+E(x_2)^2Var(x_1)+\\&E(x_1)^2Var(x_2)\end{aligned} \tag{D2}$$

附录E　矩阵的类积和

定义2　矩阵的类积和

设$\boldsymbol{A}=(a_{ij})$是一个$m\times s$矩阵，$\boldsymbol{B}=(b_{ij})$是一个$s\times n$矩阵，那么规定矩阵$\boldsymbol{A}$与矩阵$\boldsymbol{B}$的类积和是一个$m\times n$矩阵$\boldsymbol{C}=(c_{ij})$，其中

$$c_{ij}=\sum_{k=1}^{s}(a_{ik}+b_{kj}) \tag{E1}$$

式中，$i=1,2,\cdots,m$；$j=1,2,\cdots,n$，并将其记做

$$\boldsymbol{C}=\boldsymbol{A}\oplus\boldsymbol{B} \tag{E2}$$

由定义可知，

$$\boldsymbol{A}\oplus\boldsymbol{B}=\mathrm{sum}(\boldsymbol{A},\mathrm{axis}=-1,\mathrm{keepdim}=\mathrm{True})+\mathrm{sum}(\boldsymbol{B},\mathrm{axis}=-2,\mathrm{keepdim}=\mathrm{True})$$