# 3efd64016a7238741e17ffca24e84606

# 人工智能

# Transformer

# 调研报告

小组成员：王子阳、杨斌、张午潇、张子洋

2024 年 6 月 6 日

## 简介

本篇报告由小组合作而成，基于Transformer面试题进行分工，王子阳负责1-8题，对transformer进行入门介绍。杨斌负责9-11这三个题目，并拓展一些SSM，残差结构以及应用，多模态的应用。张午潇负责12-15题，包含多头注意力机制，LayerNorm的拓展知识。张子洋负责文档整合，以及第四部分的Transformer研究报告，关于Transformer+CNN结合的科研方向进行介绍。

报告中可能有些内容重合，但为确保文章连续性，不做过多删减。

目录

[简介 1](#_Toc3269)

[模块一、Transformer介绍（1-8题） 3](#_Toc19249)

[1.介绍Transformer和ViT 3](#_Toc31431)

[2.介绍Transformer的QKV 4](#_Toc3681)

[3.介绍Layer Normalization 6](#_Toc943)

[4.Transformer训练和部署技巧 7](#_Toc3556)

[5.介绍Transformer的位置编码 9](#_Toc415)

[7.介绍Transformer的Encoder模块和Decoder模块（包含第八题） 11](#_Toc15692)

[模块二、进行Transformer对比 13](#_Toc7127)

[9. Transformer和Mamba（SSM）的区别 13](#_Toc2292)

[10. Transformer中的残差结构以及意义 14](#_Toc12049)

[11.为什么Transformer适合多模态任务？ 16](#_Toc7498)

[模块三、Transformer的特点 18](#_Toc30241)

[12. 并行化 18](#_Toc29064)

[13.Transformer与LayerNorm 20](#_Toc23614)

[14. 多头注意力机制 21](#_Toc31048)

[15. Dropout技术 22](#_Toc28377)

[模块四、研究方向的应用与介绍 24](#_Toc15244)

[应用前景 24](#_Toc21234)

[基于CNN+transformer在CV方向的研究 24](#_Toc17203)

## 模块一、Transformer介绍（1-8题）

### 1.介绍Transformer和ViT

Transformer（注意力机制）：

Transformer是一种基于注意力机制的神经网络模型，用于处理序列到序列（sequence to sequence）的任务。它最初被用于机器翻译任务，但后来被证明在各种NLP任务中都非常有效。Transformer模型由编码器（Encoder）和解码器（Decoder）组成，其中编码器和解码器均由多层的自注意力机制和前馈神经网络组成。

自注意力机制让模型能够同时考虑输入序列中的所有位置，而不是仅仅依赖于固定长度的滑动窗口。它根据每个位置与其他位置的相对重要性，为每个位置计算一个权重向量，然后将每个位置的信息进行加权求和。这种注意力机制的引入使得Transformer在捕捉长距离依赖关系和处理上下文信息方面具有优势。

ViT（Vision Transformer）：

ViT是Transformer模型在计算机视觉任务上的扩展应用。传统的计算机视觉任务通常使用卷积神经网络（CNN），而ViT采用了Transformer模型来处理图像。它将图像数据切分为一系列均匀的图块（patches），将每个图块作为序列输入给Transformer模型进行处理。

ViT首先使用一个线性映射将图块转换为特征向量，然后将这些特征向量输入Transformer编码器。通过自注意力机制，ViT可以捕捉图像中不同图块之间的全局关系，并在编码器输出后接一个MLP（多层感知机）进行分类任务。ViT通过将图像划分成序列来解决传统CNN模型中全连接层带来的计算量过大的问题

区别和联系：

Transformer是一种通用的序列到序列模型，用于处理NLP任务。它的设计主要用于处理文本数据，但可以扩展到其他领域。

ViT是将Transformer模型扩展到计算机视觉领域的模型。它通过将图像划分为序列，并使用Transformer编码器来处理。

### 2.介绍Transformer的QKV

Transformer中的Q、K和V是指在自注意力机制（self-attention mechanism）中使用的三个输入表示向量。

Q表示查询向量，K表示关键向量，V表示数值向量。这三个向量是通过线性变换从原始输入向量（通常是词嵌入表示）得到的。

在自注意力机制中，以查询向量Q为基础，通过计算查询向量与所有关键向量K之间的相似度，得到一个权重分布，用于加权求和关联的数值向量V。

Q、K、V概念来源于检索系统，其中Q为Query、K为Key、V为Value。可以简单理解为Q与K进行相似度匹配，匹配后取得的结果就是V。举个例子我们在某宝上搜索东西，输入的搜索关键词就是Q，商品对应的描述就是K，Q与K匹配成功后搜索出来的商品就是V。

Transformer中，注意力的核心公式是

IMG_256

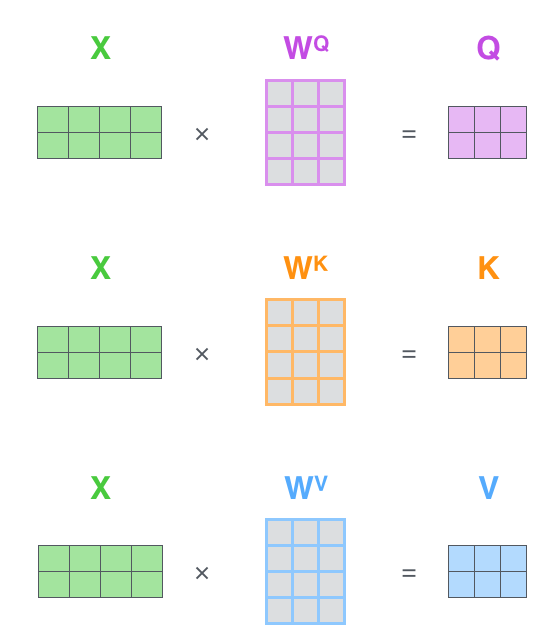
那Q、K、V是怎么来的呢？这里其实是通过对输入矩阵X进行线性变换得到的，用公式可以简单写成以下：

IMG_256

IMG_256

IMG_256

用图片直观表示为：

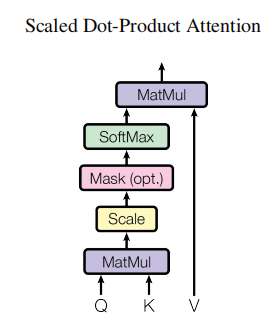


其中IMG_256、IMG_256、IMG_256是三个可训练的参数矩阵，输入矩阵X分别与三个矩阵参数进行相乘，相当于进行一次线性变换，得到了Q、K、V。

然后使用Q、K、V计算注意力矩阵，公式如下：

IMG_256

论文中给出的图如下：



Q和IMG_256经过MatMul，生成了相似度矩阵。对相似度矩阵每个元素除以IMG_256，IMG_256为IMG_256的维度大小。这个除法被称为Scale。当IMG_256很大时，IMG_256的乘法结果方差变大，进行Scale可以使方差变小，训练时梯度更新更稳定。然后经过SoftMax，最后与V做一个MatMul操作得到结果。

### 3.介绍Layer Normalization

层规范化（Layer Normalization）是一种在深度学习中常用的规范化技术，由 Jimmy Lei Ba 和 Jamie Ryan Kiros 等人于 2016 年提出。它的主要目的是帮助神经网络更快、更稳定地收敛。层规范化与其它规范化技术，如批量规范化（Batch Normalization）和组规范化（Group Normalization），有着相似的目的，但操作方式略有不同。

层规范化的工作原理如下：

（1）操作对象：层规范化在单个样本的层级上进行操作，而非在批量（Batch）的层级。这意味着它对每个样本独立计算规范化参数（均值和标准差），而不是在整个批量的基础上。

（2）计算方式：对于给定的样本，在其所有特征（或通道）上计算均值和标准差。然后，使用这些统计数据来规范化该样本的每个特征。

（3）优点：层规范化特别适用于批量大小很小或变化较大的情况，因为它不依赖于批量大小。这使得它在处理诸如在线学习或强化学习等任务时非常有用，这些任务可能无法提供稳定的批量大小。

层规范化（Layer Normalization）对于给定样本的所有特征（或通道）进行规范化的具体步骤如下：

（1）计算均值和标准差：

对于给定样本中的所有特征，计算这些特征的均值（mean）和标准差（standard deviation）。 （2）规范化特征：

使用计算得到的均值和标准差来规范化每个特征。

（3）重新缩放和偏移：

通常，在规范化之后会引入两个可学习的参数：缩放因子（scale factor）γ和偏移因子（shift factor）β。这两个参数允许模型学习规范化数据的最佳表示。

这个过程确保了每个样本的特征都具有大致相同的均值和方差，从而帮助神经网络更容易学习和快速收敛。在实际的深度学习模型中，这些步骤通常都是自动执行的，其中γ和β是通过训练过程学习得到的。通过这种方式，层规范化可以提高网络对输入数据的规模和偏移的不变性，有助于改善模型在训练和泛化方面的表现。

### 4.Transformer训练和部署技巧

**训练技巧：**

（1）数据准备和预处理：

清洗和标准化：确保数据干净且一致。对于文本数据，移除噪音字符、处理缺失值等。对于图像数据，进行归一化和数据增强。

Tokenization（对于NLP）：使用适当的分词器，如BERT的WordPiece，GPT-3的Byte Pair Encoding (BPE) 等。

（2）模型初始化和预训练：

预训练模型：使用预训练的Transformer模型（如BERT、GPT、ViT）进行微调，而不是从头开始训练，这可以大幅减少训练时间和提高性能。

权重初始化：对于从头训练的模型，使用合适的权重初始化方法，如Xavier初始化或Kaiming初始化。

（3）优化器和学习率调度：

优化器：常用的优化器包括Adam和AdamW。AdamW在处理大规模数据和复杂模型时表现出色。

学习率调度：使用动态学习率调度器（如学习率热身（warm-up）和学习率衰减策略）来控制训练过程中的学习率。线性学习率热身和余弦退火是常见的策略。

（4）正则化和归一化：

正则化：使用Dropout、L2正则化等方法来防止过拟合。

归一化：Layer Normalization在Transformer中非常重要，确保每一层的输出稳定。

（5）数据增强（对于CV）：

进行数据增强（如随机裁剪、旋转、颜色抖动等）来增加训练数据的多样性，提高模型的泛化能力。

（6）分布式训练和混合精度训练：

分布式训练：使用多GPU或多节点进行分布式训练，以处理大规模数据和加快训练速度。

混合精度训练：通过使用半精度浮点数（FP16）来加速训练过程，同时减少显存使用。

**部署技巧：**

（1）模型压缩和加速：

量化：将模型权重从32位浮点数转换为8位整数，以减少模型大小和加速推理。

剪枝：移除不重要的权重或神经元，以减小模型大小和提高推理速度。

知识蒸馏：使用大模型（教师模型）来训练小模型（学生模型），以保持性能的同时减少模型复杂度。

（2）优化推理效率：

批处理推理：在推理时处理多个输入数据，以提高计算效率。

图优化：使用深度学习框架（如TensorFlow、PyTorch）的图优化工具进行图优化，提高推理速度。

硬件加速：利用GPU、TPU或专用加速器（如NVIDIA TensorRT、Intel OpenVINO）进行推理加速。

（3）模型部署工具和平台：

使用框架提供的部署工具，如TensorFlow Serving、TorchServe、ONNX Runtime等。

部署到云平台（如AWS、Google Cloud、Azure）或边缘设备，根据需求选择合适的部署环境。

（4）监控和维护：

监控模型性能：在生产环境中监控模型的推理时间、资源使用率和预测准确性等指标。

在线学习和更新：根据新数据和反馈，定期更新模型，确保模型性能不会随着时间的推移而下降。

### 5.介绍Transformer的位置编码

**位置编码的目的**

注意力Attention这种操作具有排列不变性，输入元素位置的变动不会对注意力结果产生影响，从而模型无法感知位置信息，而在自然语言处理场景，字/词的顺序位置关系信息尤为重要，同样的字词不同的顺序可能导致句子的语言完全不一样。  
 Transformer中采用Self-Attention，每个词和整个句子所有词两两一对计算相似度权重，词与词的位置随意变动一下不会导致最终计算该对词的注意力权重产生变化，进一步导致注意力层的整个结果不变，严格来说是结果中每一个词向量计算完全一致，仅仅是词向量在输出矩阵的排列随着词和词位置互换而对应调整了一下，举个例子:

[我，爱，你] => self attention layers =>[ [0.3, 0.2, 0.1, 0.5]

[0.1, -0.1, -0.2, 0.3],

[0.5, 0.5, 0.1, -0.1] ]

[你，爱，我] => self attention layers =>[ [0.5, 0.5, 0.1, -0.1]

[0.1, -0.1, -0.2, 0.3],

[0.3, 0.2, 0.1, 0.5] ]

将[我，爱，你]输入一组Q，K，V组成的Self-Attention产出的向量，和[你，爱，我] 输入同一组Q，K，V组成的Self-Attention产出的向量，两者的结果每个词/字的embedding输出一致，仅仅是在矩阵的位置调换了一下（1和3对调）。  
 而此时如果下游网络是聚合池化操作，则池化后的结果完全一致，彻底失去位置信息，比如transformer之后使用均值池化去做文本分类，如果不做池化做flatten输出，也仅仅是特征列和特征列交换位置，对下游是全连接Dense结构而言是一致的效果。  
 因此需要引入位置编码表达出每个句子中字/词的位置信息，配合字/词本身的embedding一起加入模型进行训练。

**位置编码的多种方式**

位置编码是需要设计的，主要有绝对位置编码和相对位置编码。

绝对位置编码：在输入层做文章，为每个输入的字/词增加一个对应位置编码，该位置编码只与位置k相关，每个字/词的输入是自身编码和位置编码的融合。绝对位置编码包括可学习式，固定式等方法，比如BERT、GPT采用可学习式将位置编码当作可训练参数。

相对位置编码：在模型网络层做文章，使得模型的Self-Attention能够考虑词和词之间的相对距离，而非每个词都单独标注位置和距离，让模型通过数据自己学习位置信息。

### 7.介绍Transformer的Encoder模块和Decoder模块（包含第八题）

Transformer的架构包括两个主要模块：编码器（Encoder）和解码器（Decoder）。这两个模块协同工作，特别适合用于机器翻译等序列到序列的任务。下面详细介绍这两个模块的组成部分及其功能。

**编码器模块（Encoder）**

编码器模块由多个相同的层（通常为6层）堆叠而成。每一层包括两个主要子层：多头自注意力机制（Multi-Head Self-Attention Mechanism）、前馈神经网络（Feed-Forward Neural Network, FFNN）。此外，每个子层后还包括残差连接和层归一化。以下是编码器层的详细结构：

（1）多头自注意力机制

自注意力（Self-Attention）：自注意力机制允许每个输入元素（词或图像块）根据与序列中其他所有元素的关系来重新计算自身的表示。

多头注意力（Multi-Head Attention）：多头注意力通过并行的多个自注意力头来捕捉不同的特征和关系。每个注意力头有独立的Query(Q)、Key(K)和Value(V)矩阵。多头注意力的输出通过拼接和线性变换得到。

（2）前馈神经网络

每个编码器层还包括一个前馈神经网络，由两个线性变换层和一个ReLU激活函数组成。

（3）残差连接和层归一化

残差连接（Residual Connection）：在每个子层的输出和输入之间添加残差连接，有助于缓解梯度消失问题。

层归一化（Layer Normalization）：对残差连接的输出进行归一化，确保输入的稳定性。

**解码器模块（Decoder）**

解码器模块也由多个相同的层（通常为6层）堆叠而成。每一层包括三个主要子层：多头自注意力机制（Multi-Head Self-Attention Mechanism）、编码器-解码器多头注意力机制（Multi-Head Encoder-Decoder Attention Mechanism）、前馈神经网络（Feed-Forward Neural Network, FFNN）。同样，每个子层后包括残差连接和层归一化。以下是解码器层的详细结构：

（1）多头自注意力机制

与编码器相同，但在解码器中，为了保证生成的输出序列中的第i个位置只能依赖于之前的i-1个位置，需要进行掩码（Masking），防止看到未来的信息。

（2）编码器-解码器多头注意力机制

这个子层的目的是让解码器在生成输出序列时能够参考编码器生成的上下文表示。与自注意力类似，但这里的Query来自解码器的输入，而Key和Value来自编码器的输出。

（3）前馈神经网络

与编码器中的前馈神经网络相同，由两个线性变换层和一个ReLU激活函数组成。

（4）残差连接和层归一化

同编码器一样，在每个子层后添加残差连接和层归一化。

Transformer的编码器模块和解码器模块通过多个层的堆叠、注意力机制、前馈神经网络、残差连接和层归一化等组件的组合，能够高效地处理序列数据，并且在自然语言处理和计算机视觉等任务中取得了巨大的成功。编码器模块专注于对输入序列进行编码，捕捉其特征和关系，而解码器模块则结合编码器的输出，生成目标序列。

## 模块二、进行Transformer对比

### Transformer和Mamba（SSM）的区别

1. 架构设计与计算复杂度

Transformer 以其自注意力机制为核心，通过多头注意力（Multi-Head Attention）对输入序列进行全局建模，捕捉[长程依赖关系](https://www.zhihu.com/search?q=%E9%95%BF%E7%A8%8B%E4%BE%9D%E8%B5%96%E5%85%B3%E7%B3%BB&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22answer%22%2C%22sourceId%22%3A3477930624%7D" \t "_blank)。Transformer的架构包括编码器-解码器结构（在翻译任务中）或仅编码器结构（在语言模型中），每个编码器或解码器由多层组成，每层包含自注意力模块和[全连接前馈网络](https://www.zhihu.com/search?q=%E5%85%A8%E8%BF%9E%E6%8E%A5%E5%89%8D%E9%A6%88%E7%BD%91%E7%BB%9C&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22answer%22%2C%22sourceId%22%3A3477930624%7D" \t "_blank)（MLP）。Transformer的计算复杂度与输入序列长度的平方成正比，这是由于自注意力运算的全连接性质导致的。

Mamba结合了[状态空间模型](https://www.zhihu.com/search?q=%E7%8A%B6%E6%80%81%E7%A9%BA%E9%97%B4%E6%A8%A1%E5%9E%8B&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22answer%22%2C%22sourceId%22%3A3477930624%7D" \t "_blank)（SSM）与Transformer的MLP块，形成一种包含选择性状态空间的简单、同质架构。Mamba通过某种方式克服了Transformer在处理长序列时的复杂度瓶颈，据报道其推理速度比Transformer提升高达5倍。这表明Mamba在架构层面引入了改进，使得模型能够在保持或提高性能的同时，显著降低对序列长度敏感的计算成本。

2. 长序列处理能力

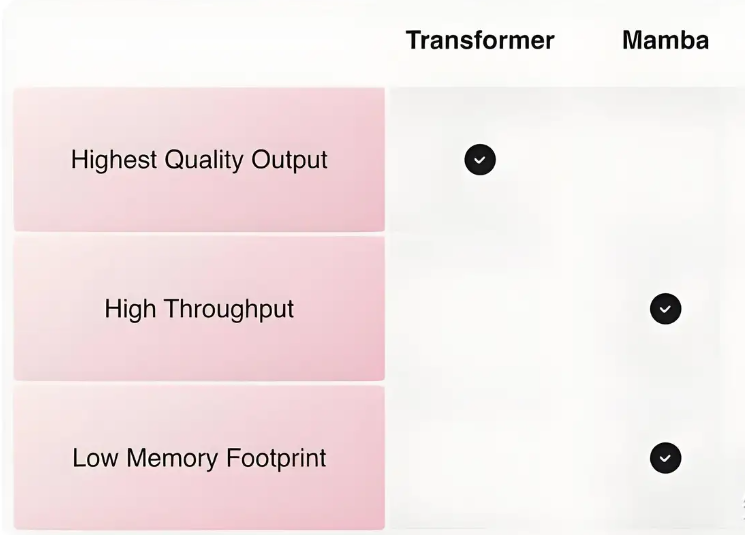
尽管Transformer通过[自注意力机制](https://www.zhihu.com/search?q=%E8%87%AA%E6%B3%A8%E6%84%8F%E5%8A%9B%E6%9C%BA%E5%88%B6&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22answer%22%2C%22sourceId%22%3A3477930624%7D" \t "_blank)能够捕获长距离依赖，但随着序列长度增加，其计算资源需求迅速增长，这限制了其在[超长序列](https://www.zhihu.com/search?q=%E8%B6%85%E9%95%BF%E5%BA%8F%E5%88%97&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22answer%22%2C%22sourceId%22%3A3477930624%7D" \t "_blank)上的应用。Mamba显然在长序列处理上具有优势，不仅能够处理更长的序列，而且在性能上与规模远大于自身的Transformer模型相抗衡。这意味着Mamba在处理文本摘要、文档理解和[大规模语料分析](https://www.zhihu.com/search?q=%E5%A4%A7%E8%A7%84%E6%A8%A1%E8%AF%AD%E6%96%99%E5%88%86%E6%9E%90&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22answer%22%2C%22sourceId%22%3A3477930624%7D" \t "_blank)等需要处理长文本的任务中，可能展现出更高的实用性和效率。

3. 可解释性与可调整性

关于Mamba的可解释性，现有的信息提及它在这方面优于Transformer，但具体如何实现未给出详细说明。可能的原因包括Mamba的架构简化、状态空间模型的内在特性，或者采用了额外的解释性技术。更好的可解释性有助于研究人员理解模型决策过程，对于模型调试、错误分析以及与人类使用者的交互至关重要。

4. 硬件优化与部署

虽然没有详细信息表明Mamba在硬件优化方面的具体措施，但其高推理速度暗示可能存在针对现代硬件（如GPU、TPU）的高效实现。高效的硬件部署意味着更低的运行成本和更快的服务响应，这对于工业级应用和大规模部署至关重要。



### 10. Transformer中的残差结构以及意义

残差网络

首先我们了解下什么是残差网络，残差网络（Residual Network，简称ResNet）是一种深度卷积神经网络（CNN）架构，由Microsoft Research Asia的Kaiming He等人在2015年提出。ResNet的核心思想是通过引入“残差学习”（residual learning）来解决深度神经网络训练中的退化问题（degradation problem）。

在传统的深度神经网络中，随着网络层数的增加，理论上网络的表示能力应该更强，但实际上，过深的网络往往难以训练，性能反而不如层数较少的网络。这种现象被称为“退化问题”，即随着网络深度的增加，网络的准确率不再提升，甚至下降。

ResNet通过引入“跳跃连接”（skip connections）或“捷径连接”（shortcut connections）来解决这个问题。在ResNet中，输入不仅传递给当前层，还直接传递到后面的层，跳过一些中间层。这样，后面的层可以直接学习到输入与输出之间的残差（即差异），而不是学习到未处理的输入。这种设计允许网络学习到恒等映射（identity mapping），即输出与输入相同，从而使得网络可以通过更简单的路径来学习到正确的映射关系。

在Transformer模型中，残差网络的使用主要是为了解决自注意力机制（self-attention）带来的问题。Transformer模型完全基于注意力机制，没有卷积层，但其结构本质上也是深度网络。在Transformer中，每个编码器（encoder）和解码器（decoder）层都包含自注意力和前馈网络，这些层的参数量非常大，网络深度也很容易变得很深。

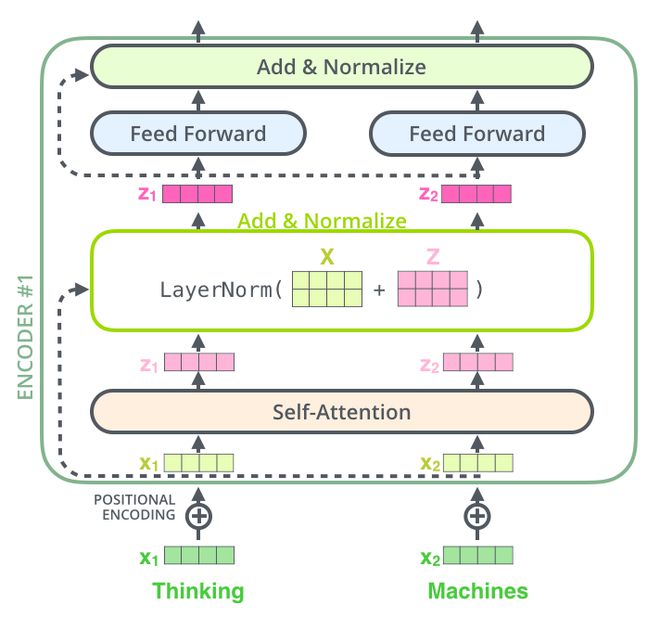
使用残差连接可以帮助Transformer模型更有效地训练深层网络。在Transformer的自注意力层中，输入通过自注意力和前馈网络后，与原始输入相加，形成残差连接。这种设计使得网络即使在增加更多层数时，也能保持较好的性能，避免了退化问题。

总结来说，残差网络在Transformer模型中的应用解决了以下几个问题：

* **缓解退化问题：通过残差学习，使得网络即使在增加层数时也能保持或提升性能。**
* **加速收敛：残差连接提供了梯度的直接路径，有助于梯度在深层网络中的传播，加速训练过程。**
* **提高表示能力：允许网络学习更复杂的函数，同时保持对简单函数的学习能力。**

Transformer模型的成功部分归功于残差连接的设计，这使得它能够构建更深、更强大的模型，从而在自然语言处理（NLP）和计算机视觉等领域取得了显著的成果。

可以使用下面这张图来解释残差网络，原始向量x在经过自注意力层之后得到z向量，为了防止网络过深带来的退化问题，Transformer模型使用了残差网络，具体做法是使用计算得到的z矩阵，在和原始输入的x矩阵做残差链接，即图中的X+Z，然后使用LayerNorm函数进行层归一化，计算得到新的z向量，然后输入到前馈层。



### 11.为什么Transformer适合多模态任务？

多模态（multimodal）是指涉及到多种模态（如视觉、语音、文本等）的数据或信息。在计算机科学和人工智能领域中，多模态通常指将多种类型的数据或信息相结合，来解决特定的问题或任务。

以图像识别为例，图像可以被视为一种视觉模态，而对图像的分类或识别就是单模态任务。而如果将图像和语音或文本数据相结合，就可以做更复杂的任务，如图像描述（image captioning）或视觉问答（visual question answering），这些任务需要同时利用图像和语言模态的信息。

transformer在多模态领域的强大，是因为它自身的self-attention结构能适应各种不同类型的数据，使得各种数据在模式的对齐上表现更加优秀。Transformer中抛弃了传统的CNN和RNN， 整个网络结构完全是由Attention机制组成。更准确地讲，Transformer由且仅由self-Attenion和Feed Forward Neural Network组成。

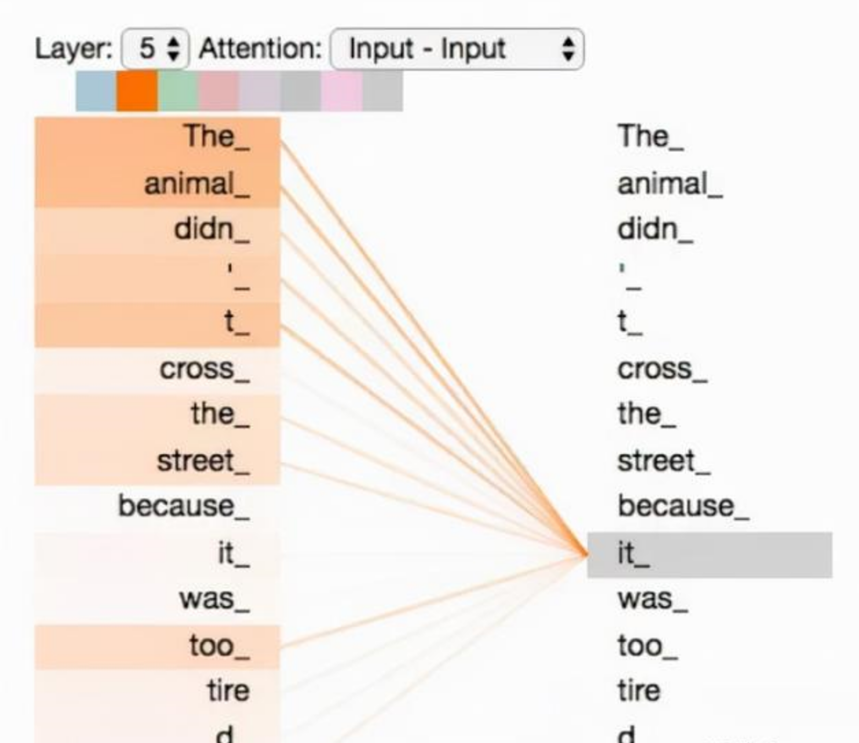
首先，我们从self-attention机制说起。该机制具有非常强大的特征提取功能。

正如ViT，他只是将一张图片均匀的切成九个patch，将每个patch拉成一维向量，在传入self-attention机制中，模型能通过该机制将九个patch内的信息进行交互，从而知道哪个patch比较重要，每个patch之间有何联系等，具体如下图所示。



ViT的attention机制找到了图片中最有意义的patch，从而得到了好的结果。

类似的，在其他地方， 如nlp，表格数据方面，它们一般本身是一维长序列数据， transformer同样能很好地处理他们，如下图 的nlp数据为例，



将文本数据输入模型，通过观察直线的粗细，可以发现，模型认为it指代的是这句话的animal这个名词。以上两个例子均说明了transformer在各个领域都具有优秀的表现。

## 模块三、Transformer的特点

### 12. 并行化

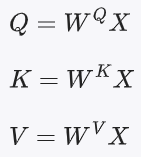
1.自注意力机制的并行化

在传统的RNN或LSTM模型中，序列数据（例如句子）需要逐步处理，每一步计算都依赖于前一步的结果，因此无法并行处理。这种顺序依赖性导致计算效率低下，尤其是在处理长序列时。

而在Transformer中，自注意力机制允许模型同时处理输入序列中的所有位置。具体来说，自注意力机制通过以下步骤实现并行化：

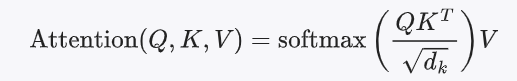
（1）自注意力机制 (Self-Attention)

计算键（Key）、查询（Query）和值（Value）：每个输入词向量通过三个不同的线性变换生成键、查询和值向量，这些操作可以在所有输入单词上同时进行，因此可以并行化。



其中Q,K和W分别是查询、键和值矩阵，X是输入词向量矩阵。

（2）计算注意力得分：对于每一个查询向量，计算其与所有键向量的点积类似度，得到注意力得分矩阵。这个步骤也可以在所有输入位置上并行计算：



其中，dk是键向量的维度，softmax操作也是在所有计算结果上并行进行的。

（3）加权求和：注意力得分矩阵与值向量矩阵相乘，得到每个输入位置的加权求和结果，这个步骤同样可以并行处理。

由于这些步骤都可以在矩阵的维度上进行并行计算（非常适合GPU等硬件的矩阵运算），大大提高了计算效率。

2.多头自注意力机制

Transformers通过使用多头注意力机制进一步利用并行计算的优势。多头注意力机制将注意力计算扩展为多个头，每个头独立地执行上面描述的自注意力计算，最终将结果拼接在一起并进行一次线性变换。

多头注意力机制的步骤：

（1）多个头的计算：每个头都有自己的一套权重矩阵，这样可以在多个不同的“视角”上独立进行注意力计算。

（2）并行执行：所有这些头的计算可以同时进行，因为它们是相互独立的。

（3）拼接与线性变换：计算出的多头注意力结果在维度上拼接起来，然后通过一个线性变换得到最终的输出。

3. 前馈神经网络的并行化

在每一个自注意力层之后，Transformer都有一个前馈神经网络（Feed-Forward Neural Network, FFNN），它由两个全连接层组成：



全连接层的计算：

第一层：输入每个词向量，并行计算线性变换和非线性激活函数ReLU。

第二层：对ReLU输出的结果再次进行线性变换。

因为这些操作都是逐位置（position-wise）的操作，所以可以对所有输入位置同时执行，即并行处理。

4.串行依赖性的消除

传统的RNN/LSTM由于其序列化的本质，导致每个时间步的输出需要依赖前一步的输出。因此，无法并行计算。而Transformer的架构完全消除了这种串行依赖性，使得每一层（无论是自注意力层还是前馈神经网络层）都能独立地对整个输入序列进行操作，大大加速了训练和推理过程，并且利用了现代硬件（如GPU和TPU）的并行计算能力。

### 13.Transformer与LayerNorm

Transformer中一般使用LayerNorm（层归一化）的主要原因是为了稳定训练过程，提高模型性能。

1.稳定训练过程

神经网络在训练过程中，层的输入分布可能会发生变化，这是所谓的“内部协变量偏移”（Internal Covariate Shift），这种变化会导致模型的训练变得不稳定。LayerNorm通过在每一个迷你批次的训练数据中对每一层的输入进行归一化，使输入数据的分布保持稳定，从而缓解了这种问题，确保模型在训练过程中能够更快收敛。

2. 应对深层网络的梯度消失和梯度爆炸问题

在深层神经网络中，梯度消失和梯度爆炸是两个常见问题。LayerNorm通过归一化处理，可以在一定程度上缓解这些问题，使梯度在反向传播过程中保持较为稳定，避免梯度爆炸或消失，从而有利于深层网络的训练。

2.提高模型的泛化能力

LayerNorm不仅对训练过程有益，还可以提升模型的泛化能力。归一化处理使得模型对每一批次数据的依赖性减弱，能够更好地适应不同的数据分布，从而在测试数据上表现得更好。

3.与BatchNorm的对比

虽然BatchNorm（批归一化）也是一种常用的归一化方法，但它在应用于Transformer时有一些局限性：

（1）序列数据处理：BatchNorm通常在图像处理任务中表现良好，但对于自然语言处理（NLP）中的序列数据，尤其是变长序列，BatchNorm的表现可能不稳定。而LayerNorm在处理变长序列时更加稳定和适应。

（2）小批量数据：BatchNorm在小批量数据训练时，统计量可能不稳定，而LayerNorm不依赖于批大小，只对当前层的输入进行归一化，因此在小批量数据上也表现良好。

4.Transformer的LayerNorm实现

在Transformer中，LayerNorm通常在以下两个位置使用：

（1）在子层内部的归一化：包括多头自注意力机制和前馈神经网络（FFNN）之间。LayerNorm的应用有助于在每个子层的输入上进行归一化处理，使梯度传播更加稳定。

（2）残差连接后的归一化：在每个子层计算出结果后，将其与输入进行残差连接（Resitual Connection），这种连接方式可以在深层网络中更好地传播梯度。LayerNorm通常放在残差连接之后，进一步稳定输入的分布。

### 14. 多头注意力机制

1. 提供多视角的表示能力

单头注意力机制只能在一次计算中捕获一种类型的关联关系，这限制了模型的表示能力。而多头注意力机制通过引入多个独立的注意力头，使模型能够在不同的子空间中捕获不同类型的依赖关系，从而提供更丰富的表示。

2.增强模型的表达能力

通过使用多头注意力机制，模型可以并行地关注到输入序列中的不同位置和模式。例如，一个头可能专注于短程依赖关系，而另一个头可能专注于长程依赖关系。这种多样化的信息捕捉方式使得模型在同一层中能够同时处理多种依赖关系，使表达能力更强。

3. 改善梯度流动，稳定训练过程

多头注意力机制将注意力计算分散到多个独立的头上，使得每个头的计算负担较轻，从而改善了梯度流动的稳定性。每个头的输出经过拼接和线性变换后，进一步融合了多视角的信息，有助于模型以更稳定的方式进行训练。

4. 提高并行计算效率

多头注意力机制允许多个注意力头在同一时间并行计算，由于现代硬件（如GPU和TPU）非常适合处理这种并行计算，因此多头注意力机制能够在提升表达能力的同时不显著增加计算开销。反而因并行化操作显著提高了计算效率。

5. 减少单个注意力头的计算复杂度

由于单个注意力头的计算复杂度较高，直接处理整合后的高维表示可能会受到计算资源的限制。多头注意力机制将整体的计算任务分解为多个头的并行计算，每个头只需要处理较低维度的查询、键和值向量，从而降低了单个注意力的计算复杂度。

### Dropout技术

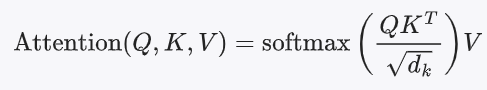
在Transformer模型的训练过程中，Dropout是一种常用的正则化技术，用于防止过拟合并增强模型的泛化能力。

1. 自注意力机制中的Dropout

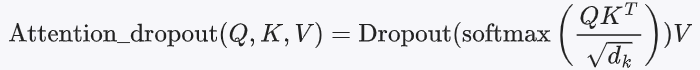
在自注意力机制的计算过程中，包括计算注意力权重和应用注意力权重时，通常会使用Dropout。

（1）注意力权重的计算：

在计算注意力权重时，首先需要用softmax函数对权重进行归一化，此时可以在softmax函数输出的权重上应用Dropout。



在上面的公式中，softmax后的结果会乘以一个随机的Dropout mask，以随机丢弃一部分权重：



（2）值向量的Dropout：

计算得到的加权值向量（value）也可以应用Dropout，防止过拟合：



1. 前馈神经网络中的Dropout

在每一层的前馈神经网络（Feed-Forward Neural Network, FFN）中，也使用了Dropout。这通常会在非线性激活函数ReLU之后和最终输出之前应用。

（1）激活函数后的Dropout：

在ReLU激活函数之后，应用Dropout以防止网络过拟合：



（2）线性层后的Dropout：

在前馈网络的第二个线性层之上，应用Dropout：



1. 残差连接后的Dropout

在Transformer的架构中，每个子层都有残差连接（Residual Connection）。在残差连接计算中，也可以包含Dropout，用于进一步的正则化。

子层输出的Dropout：

针对每个子层的输出进行Dropout，然后再进行残差连接：



其中，SubLayer(x)代表自注意力机制或者前馈神经网络的输出。

1. Embedding层的Dropout

在输入层，通常会对输入的词嵌入（Embedding）进行Dropout处理，以增加训练过程的随机性，防止过拟合。

Embedding的Dropout：

对输入的词嵌入应用Dropout：



1. 参数设定

在实际的Transformer实现中，Dropout的具体概率参数通常是一个预定义的超参数，可以根据具体任务和数据集进行调整。常见的设定包括：

（1）attention\_probs\_dropout\_prob：自注意力权重的Dropout概率。

（2）hidden\_dropout\_prob：前馈神经网络层和残差连接后的Dropout概率。

（3）embedding\_dropout\_prob：输入Embedding的Dropout概率。

这些概率一般在训练过程中设定为较小的值，比如0.1或0.2，以确保模型的训练能够取得较好的正则化效果，同时不过度丢失有效信息。

## 模块四、研究方向的应用与介绍

### 应用前景

1. 自然语言处理 (NLP)

Transformer 模型在 NLP 中的应用已经非常成熟，并且其前景依然广阔。

* 机器翻译：如 Google Translate 使用 Transformer 模型提高翻译质量。
* 文本生成：如 GPT-3、GPT-4 等生成型预训练模型能够生成高质量的自然语言文本。
* 文本摘要：用于自动生成文章摘要，帮助用户快速获取文章的核心内容。
* 问答系统：如 ChatGPT，能够理解并回答用户提出的问题。
* 情感分析：通过分析用户文本的情感倾向，为企业提供用户反馈分析。

2. 计算机视觉

* 图像分类：如 Vision Transformer (ViT)，能够进行高效的图像分类任务。
* 目标检测：用于检测图像或视频中的特定目标对象。
* 图像生成：如 DALL-E，能够根据文本描述生成相应的图像。
* 图像分割：用于将图像分割成多个有意义的部分，应用于医疗图像处理等领域。

3. 语音处理

* 语音识别：如用于将语音转换为文本。
* 语音合成：如用于将文本转换为自然的语音输出。
* 语音翻译：直接将一种语言的语音翻译成另一种语言的语音。

4. 推荐系统

* 个性化推荐：通过分析用户的历史行为和兴趣，提供个性化的推荐内容。
* 协同过滤：用于在大规模用户和商品数据中进行高效的协同过滤。

5. 金融领域

* 股票预测：分析历史数据和新闻资讯，预测股票价格走势。
* 风险管理：通过大数据分析，评估和管理金融风险。
* 欺诈检测：检测和预防金融交易中的欺诈行为。

### 基于CNN+transformer在CV方向的研究

我们在文章前面已经对于CNN进行充分的说明，transformer对于CV来说据有重要的创新意义，我们接下来对于这个方面进行一个CV方向项目的介绍，基于CNN和Transformer来设计高效准确的单目标追踪算法。

本项目旨在基于CNN和Transformer来设计高效准确的单目标追踪算法，研究思路主要包含以下四个方面：

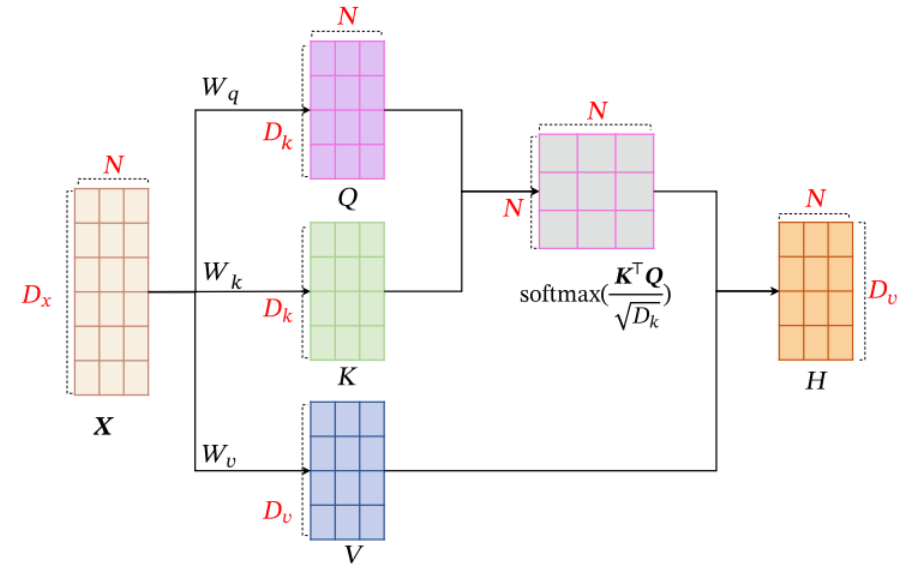
1. 设计新的CNN+Transformer模型架构来解决目CNN+Transformer单目标跟踪算法存在的卷积核感受野限制和图像高频空间信息丢失的问题；
2. 针对一些难以解决的复杂场景添加专门的模块解决；
3. 针对追踪目标随时间变化，造成传统的模板匹配方式在长距离追踪后性能下降的问题，在模型中加入历史帧提供的追踪目标外观变化信息；

第四，针对当前单目标追踪领域普遍存在的计算冗余问题，设计能够并行处理的算法框架来提高模型的运行速度。



**1.设计全新的CNN+Transformer模型架构来兼顾图像浅层和深层特征并增大感受野**

使用Transformer的编码器模块来对CNN不同层次的特征进行建模并进行特征注意力融合。CNN采用深度残差网络（Deep residual network,ResNet），将ResNet50提取图像特征的第三层和第四层输出，对这两层的输出特征分别通过不同数量的1×1卷积核来进行降维操作，之后分别输入Transformer的编码器模块，对全局的特征关联进行建模。之后，利用一个Transformer的编码器模块来对建模后的全局特征通过注意力机制进行融合，最终输出ResNeT50多层次融合后的全局特征关联信息。卷积神经网在提取特征的过程中，经过卷积、下采样和上采样后，最终的特征图丧失了追踪物体的一些细节，在一定程度上影响了追踪的精度。因此，在此基础上，基于多层特征融合的网络结构可以获得更加丰富的语义信息。



**2.设计联合辅助并行解码器来加入追踪目标随时间变化的信息**

加入前一帧目标特征作为输入的Transformer解码器模块，作为目标随时间变化的动态信息，与当前检测进行相似度计算来辅助当前帧的追踪。此部分主要分为特征提取和关联匹配两个部分，而特征提取部分分为对前一帧特征的提取和检测特征两部分。由于视频帧的连续性，当出现追踪目标消失导致所提取的前一帧特征失效的情况时，便使用检测特征为新的追踪目标的特征提取器。

此外，基于Tranformer解码器的注意力机制，利用其中的多维注意力参数，可以在模型中同时对神经网络的浅层特征和深层特征进行测试，保留单位的高频空间细节信息和目标语义信息，通过对不同特征组合测定的比较，筛选出最有决定性的一组作为考量。同时，在这个过程中使用堆叠的方式对Transformer的解码器进行排列，可有效减少参数复杂度的提升，在扩大感受野，加入浅层空间特征的同时控制模型的参数和计算复杂度不会明显增长。

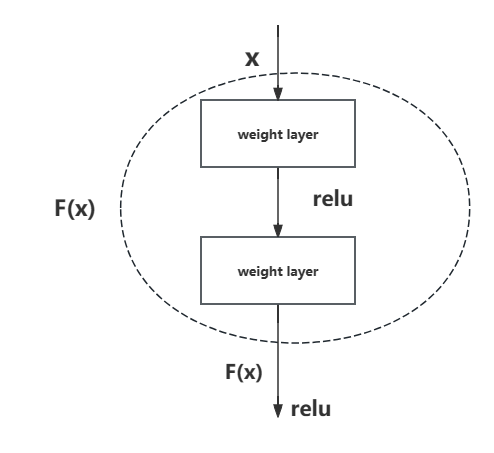
**3.通过设计多支路Tranformer解码器并行的方式提升模型的运行速度**

以Transformer的基本模型架构为基础，在网络中同时进行对目标的检索与追踪，在不降低追踪精度的基础上提升模型的运行速度。这种—段式的目标追踪范式，即检测和追踪同时进行的框架为减小模型的计算开销带来了优势，同时可以强化追踪精度与速度。

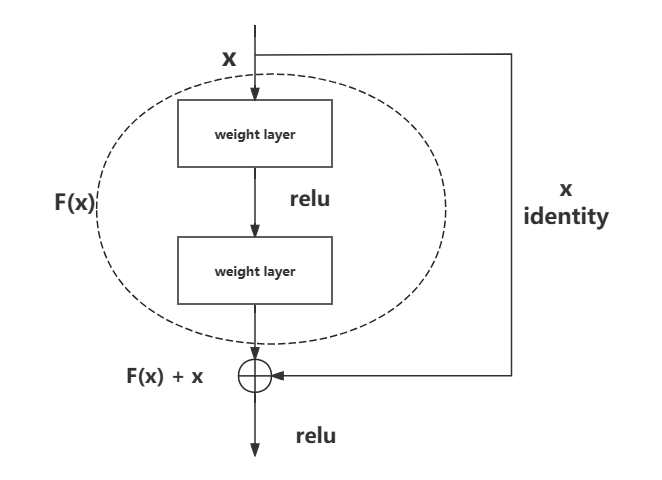
1. **卷积神经网络的创新**

在该项目中使用了深度残差网路(ResNet)对常规卷积神经网络进行深层特征优化。对于常规卷积神经网络，在训练过程中梯度不断更新,通过反向传播。随着训练层数的加深，会产生梯度弥散现象，从而无法对前面的权重做出有效调整会出现错误率不降反升的情况，即产生退化。残差网络能有效解决传统神经网络中常见的梯度消失和表示瓶颈问题，使得网络能够进行更深层次的迭代训练，从而提升了模型的性能。

让我们聚焦在神经网络局部，假设原始输入为。某一层内的最优函数即为该层内部经处理多得到得到的输出，作为下方的输出。



对于残差块而言，我们定义某一层内的最优函数，被称为残差函数，事实上，由两部分函数组成，恒等函数和残差函数。恒等函数由上一次输出的结果也就是该层输入进行恒等映射所得到。避免了负优化问题。



1. **Transformer的创新**

①多层次特征融合：通过将ResNet50的第三层和第四层的特征分别输入Transformer编码器模块，并进行特征注意力融合，可以更好地兼顾图像的浅层和深层特征，增大感受野，从而提高模型的精准度。

②自适应特征组合：利用Transformer解码器的注意力机制，可以有效融合CNN浅层特征和深层特征，调整最优的特征组合，从而更好地捕获图像中的细节信息和多层次的信息，提升模型的表现能力。

③联合空间域和通道域：在CNN主干网络中加入联合空间域和通道域的注意力机制模块，在追踪对小尺度目标或者相似目标过多时，提高模型的追踪能力以及模型在复杂场景中的精准度和应用性。

④动态信息辅助追踪：设计了联合辅助并行解码器来加入目标随时间变化的信息，利用前一帧目标特征作为输入的Transformer解码器模块来辅助当前帧的追踪，有效解决了目标尺度小和过多相似目标对识别的干扰问题。由于在视频流中，每一个前后帧的关联性较强，模型可以更有效地进行目标追踪，提高追踪的准确性。

⑤多支路并行：通过设计多支路Transformer解码器并行的方式，可以提升模型的运行速度，使得模型更加适用于实时性要求较高的场景，同时保持良好的追踪效果。

1. **总结**

项目对于CNN+Transformer的结构提出了全新的创新，用来解决CV中的单目标追踪问题，是Transformer在科研领域的一个应用，相信在未来Transformer会有更大的科研空间。