中图分类号：TP391.4

论 文 编 号 ：10006SY2015213

北京航空航天大學

硕士学位论文

孪生网络目标跟踪算法中相关运算的关键技术研究

作者姓名　　沈天琦

学科专业　　控制科学与工程

指导教师　　张弘　教授

培养学院　　宇航学院

Research on Key Technologies of Correlation Operation in Single Object Tracking with Siamese Network Structure

A Dissertation Submitted for the Degree of Master

Candidate：Shen Tianqi

Supervisor：Prof. Zhang Hong

School of Astronautics

Beihang University, Beijing, China

中图分类号：TP391.4

论 文 编 号 ：10006SY2015213

硕 士 学 位 论 文

孪生网络目标跟踪算法中相关运算的关键技术研究

作者姓名　沈天琦　申请学位级别　工学硕士

指导教师姓名　张弘　职　称　教授

学科专业　控制科学与工程　研究方向　模式识别与智能系统

学习时间自　2020 年 09 月 01 日　起至　2022 年 12 月 31 日止

论文提交日期　2022 年 11 月 21 日　论文答辩日期　2022 年 12 月 12 日

学位授予单位　北京航空航天大学　学位授予日期　2022 年 1 月 3 日

关于学位论文的独创性声明

本人郑重声明：所呈交的论文是本人在指导教师指导下独立进行研究工作所取得的成果，论文中有关资料和数据是实事求是的。尽我所知，除文中已经加以标注和致谢外，本论文不包含其他人已经发表或撰写的研究成果，也不包含本人或他人为获得北京航空航天大学或其它教育机构的学位或学历证书而使用过的材料。与我一同工作的同志对研究所做的任何贡献均已在论文中作出了明确的说明。

若有不实之处，本人愿意承担相关法律责任。

学位论文作者签名：沈天琦　　日期：　2022 年 12 月 13 日

学位论文使用授权书

本人完全同意北京航空航天大学有权使用本学位论文（包括但不限于其印刷版和电子版），使用方式包括但不限于：保留学位论文，按规定向国家有关部门（机构）送交学位论文，以学术交流为目的赠送和交换学位论文，允许学位论文被查阅、借阅和复印，将学位论文的全部或部分内容编入有关数据库进行检索，采用影印、缩印或其他复制手段保存学位论文。

保密学位论文在解密后的使用授权同上。

学位论文作者签名：沈天琦　　日期：　2022 年 12 月 13 日

指导教师签名：　　日期：　2022 年 12 月 13 日

摘　要

视频单目标跟踪是计算机视觉的基础任务之一，基于孪生网络的单目标跟踪算法凭借其优越的跟踪性能被广泛应用于航空航天、军事和民用等领域，该算法中的相关运算技术负责有效融合网络中的模板图像特征和搜索图像特征，直接影响算法能否有效利用模板中的目标信息在搜索区域中进行目标的识别与定位。

在空间维度和通道维度对多尺度的模板图像特征和搜索图像特征进行相关运算十分关键，将影响孪生网络的单目标跟踪算法能否适应跟踪过程中目标的形变。本课题探究孪生网络目标跟踪算法中相关运算的关键技术：针对空间维度特征相关运算复杂度高的问题，本课题提出了基于相位感知注意力空间特征融合的孪生网络跟踪算法。将图像特征视为一种“波”，利用相位关系自适应调整融合的权重，去掉了传统注意力算子，降低了网络的计算开销。针对通道维度特征相关性难以建模的问题，本课题设计了基于图注意力通道特征融合的孪生网络跟踪算法。将通道维度特征作为随机图上的节点特征并用图注意力卷积进行处理，成功建模了通道维度特征的相关性并对特征进行融合。针对相关运算中缺少多尺度特征融合的问题，本课题设计了基于分流注意力多尺度特征融合的孪生网络跟踪算法。基于 Transformer 通过内部设计分流自主语境增强模块和分流交叉特征增强模块，分别对多尺度注意力算子进行增强表示与特征融合。本课题设计的算法均在公开数据集上取得良好的性能评估结果。

跟踪算法应用于边缘设备具有十分重要的工业价值，本课题针对地平线旭日 X3 开发板上的硬件资源有限的情况，对算法中的算子进行优化并使用工具链完成模型的部署，搭建了实时目标跟踪系统，在实际应用场景中取得良好的跟踪效果。

关键词：单目标跟踪，孪生网络，多层感知机，图神经网络，Transformer 网络

Abstract

Video single object tracking is one of the basic tasks of computer vision. The single object tracking algorithm based on the Siamese network is widely used in aerospace, military and civil fields due to its superior tracking performance. The correlation technologies in the algorithm are responsible for effective fusion. The template image features and search image features in the network directly affect whether the algorithm can effectively use the template information of the object to identify and locate the object in the search area.

It is very important to fuse multi-scale template image features and search image features in spatial dimension and channel dimension, which will affect whether the single object tracking algorithm based on the Siamese network can adapt to the object deformation in the tracking process. This research explores the key technologies of correlation in the object tracking algorithm based on Siamese network: To solve the problem of inefficient computation of spatial dimension features' correlation operation, this research proposes a Siamese network tracking algorithm based on phase-aware attention mechanism, which fuses features in space dimension. One image feature is viewed as a wave and the weights of features when being fused are adaptively adjusted by using the phase relationship, which eliminates the traditional attention operator and reduces the computational cost of the network. To solve the problem that it is difficult to model the correlation of channel dimension features, this research designs a Siamese network tracking algorithm based on graph attention mechanism, which fuses features in channel dimension. One feature in channel dimension is viewed as a node feature on a random graph and all channel features are processed with graph attention convolution. The correlation of channel features is successfully modeled and fused. To solve the problem of lack of multi-scale feature fusion in correlation operation, this research designs a twin Siamese tracking algorithm based on shunted attention mechanism, which fuses multi-scale features. The shunted ego-context augment module and shunted cross-feature augment module designed in the transformer structure enhance and fuse multi-scale attention operators effectively. All the three algorithms achieve good performance evaluation results on public datasets.

Important industrial value can be attached to one tracking algorithm which can be applied on edge devices. In consideration of the limited hardware resources on the Horizon Sunrise X3 development board, this research optimizes the operators in the algorithm and uses the tool-chain to complete the deployment of the model. Finally, a real-time object tracking system is built successfully, which achieves good tracking performance in actual application scenarios.

Key words: Single Object Tracking, Siamese Network, Multi-Layer Perceptron, Graph Neural Network, Transformer

目 录

图清单

表清单

第一章　绪论

1.1 研究背景和意义

视频单目标跟踪作为计算机视觉领域中的一个基础研究课题[1]，是导航和制导[2]、视频监控[3]、场景理解[4]等任务的基础，可以广泛应用于航空航天[5]、军用[6]和民用[7]等领域。

如图 1 所示，得益于边缘软硬件设备的迅猛发展，搭载视频单目标跟踪算法的智能系统已经应用于各个领域：

- 航天器追踪：对进入大气层后的航天器进行跟踪测量，将实时准确的数据提供给控制中心，并为地面搜救单位提供有效目标落点数据，对航天器平安落地与相关人员的生命安全进行有效保障。
- 军事目标侦察打击：在侦察方面，相关人员对特定目标的锁定和跟踪，有助于掌握目标的动向及意图，为指挥员做出正确判断，采取正确行动提供保障；在打击方面，良好的跟踪算法，为武器系统不断提供目标信息，配合武器系统的导航制导和控制系统的运作，最终实现成功打击。
- 民用监控智能安防：结合目标跟踪技术的智能监控系统能够在海量视频数据中不断跟踪感兴趣目标，快速定位目标，为后续视频分析提供重要的辅助信息。

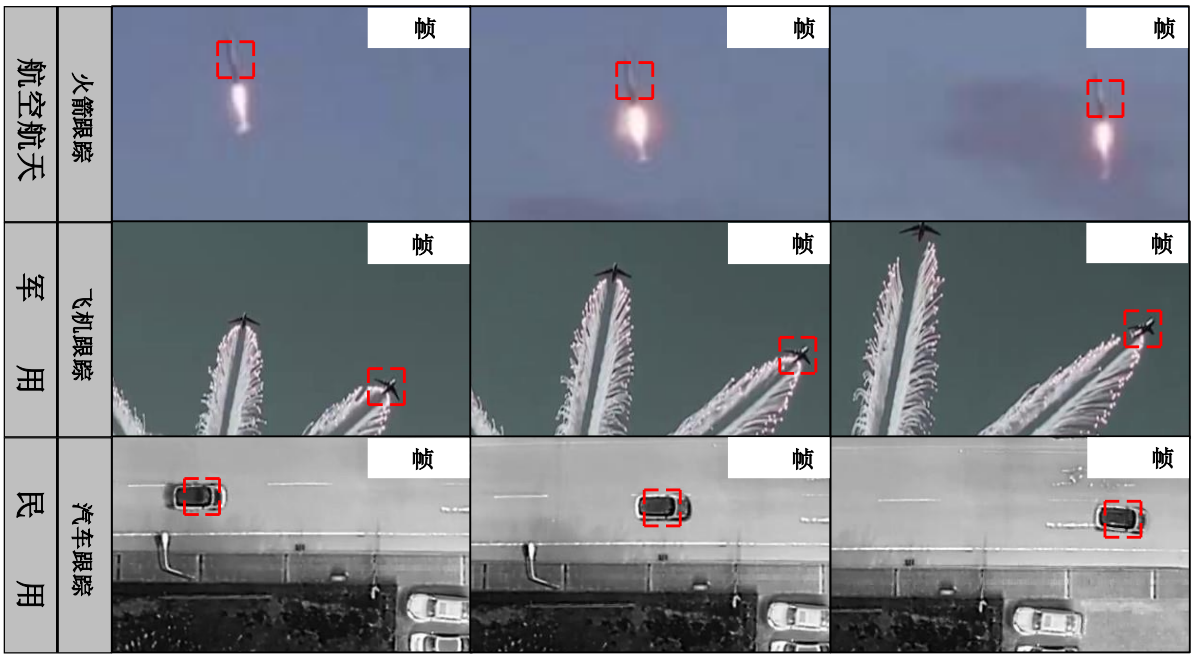

图 1　视频单目标跟踪的应用

视频单目标跟踪的任务为：针对视频序列，首先根据初始视频帧中目标的状态，将跟踪器初始化，然后提取目标特征并建立目标模型，在后续视频帧中使用设计的跟踪策略，基于目标模型估计得到目标在当前帧的状态，最后利用当前状态更新目标模型，继续下一帧的跟踪。

在视频单目标跟踪领域中，目前主流的算法主要分为两大类：基于相关滤波的单目标跟踪算法和基于孪生网络的单目标跟踪算法。其中，基于相关滤波[8]的单目标跟踪算法使用灰度特征进行目标外观表达，使用循环矩阵提取样本，并将时域的计算转换到频域，实现高效率的目标跟踪。由于灰度特征不足以表征目标，因此梯度方向直方图特征和颜色特征等特征也被加入到目标跟踪中。

然而，人工设计的特征仍不能很好地表达目标的特点。基于孪生网络[9]的视频单目标跟踪算法采用神经网络通过在数据集上的训练，学习得到更具有表达能力的特征，并将这些特征应用于搜索区域中的目标定位。如图 2 所示，常见的孪生网络目标跟踪算法的输入为两张图片，一张是待跟踪目标的模板图像，另一张是尺寸稍大的搜索区域的图像。网络需要学会如何在搜索图像中分辨出是否存在目标，如果存在，则还需用边界框（bounding box）将目标标识。

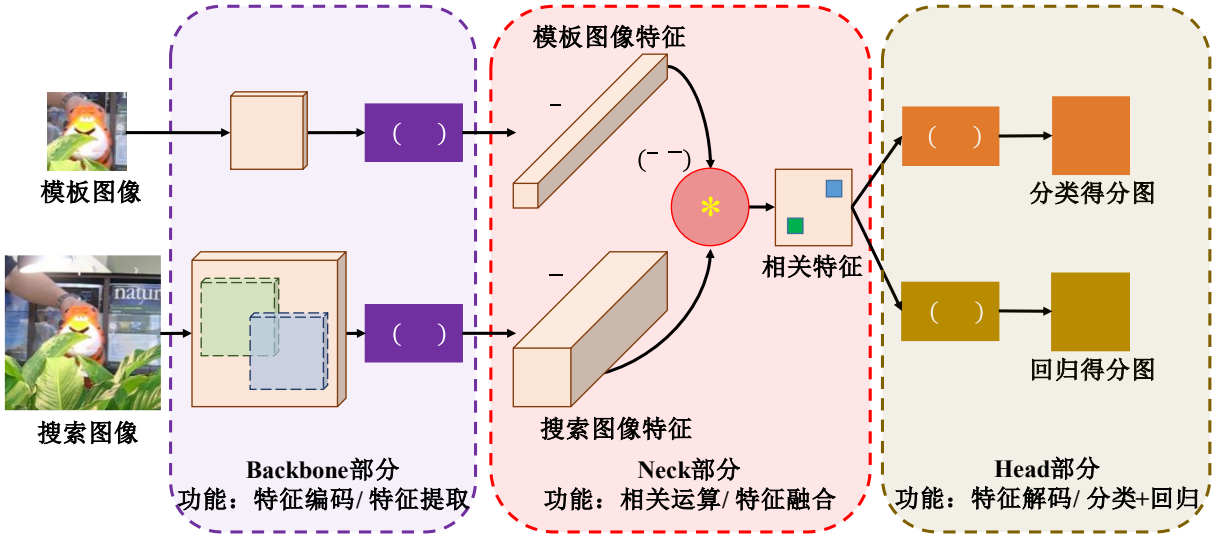

图 2　基于孪生网络的目标跟踪算法结构图

网络的结构分为三个部分：骨干（Backbone）部分，一般采用卷积神经网络（Convolution Neural Network）提取来自模板图像和搜索图像的深度特征；脖颈（Neck）部分，一般采用一些相关运算（Correlation Operation）融合模板图像特征和搜索图像特征，得到融合特征；头（Head）部分，一般采用一些简单的卷积操作将融合特征变换得

到分类得分图和回归得分图，其中分类得分图用于目标识别（判断是否存在目标），回归得分图用于目标定位（得到目标的边界框）。

在近些年的视频目标跟踪挑战赛[10]中，主流算法均为采用基于孪生网络的视频单目标跟踪算法，在短时跟踪、实时跟踪、长时跟踪等任务中都取得了很好的表现。然而，基于孪生网络的视频单目标跟踪算法，很大程度上依赖 Backbone 部分的特征提取和 Neck 部分的相关运算。其中，特征提取的好坏直接影响了提取得到的特征能否足够表征模板图像与搜索图像，而相关运算的好坏直接影响了模板图像能否有效指导搜索图像中的目标识别与定位。

在近些年的研究中已经攻克了一些技术难关[11,12]，使得更加强大的深度特征提取网络被使用在 Backbone 部分进行高效地特征提取，逐渐形成了统一的范式；而相关运算的设计并没有达成统一，如何设计高效的相关运算模块来利用模板图像特征去指导搜索图像中的目标识别与定位仍然是一个研究热点。

因此，本文针对基于孪生网络的视频单目标跟踪算法中的相关运算技术，研究如何设计高效的相关运算模块，这对改进基于孪生网络的视频单目标跟踪算法效果具有十分重要的意义。

1.2 国内外研究现状

孪生网络目标跟踪算法中相关运算的关键技术作为最近的研究热点之一，其分类目前仍没有统一的划分方式。由于相关运算与特征提取网络一样经常进行模块化设计，因此本文采用与特征提取网络一致的分类方法，将相关运算分类为：基于多层感知机的相关运算技术、基于卷积神经网络的相关运算技术、基于图神经网络的相关运算技术、基于 Transformer 的相关运算技术。

本文将会调研目前按照上述分类方法中的各相关运算技术的研究现状，分析其利用模板图像特征指导搜索图像中目标的识别与定位的合理性与有效性。其中，基于多层感知机的相关运算技术目前没有相应的研究，因此不在研究现状中总结。

1.2.1 基于卷积神经网络的相关运算技术

基于卷积神经网络的相关运算技术是最先出现的相关运算技术，其特点是实现简单，速度很快，相应的研究深入探究了不同的卷积方式。

(1)直接卷积

文献[13]使用图 3 所示的相关运算技术，其中$\bar{\boldsymbol{z}}$是模板图像经过特征提取网络所得到的模板图像特征，$\bar{\boldsymbol{x}}$是搜索图像经过特征提取网络所得到的搜索图像特征。直接卷积是将$\bar{\boldsymbol{z}}$当作卷积核，将$\bar{\boldsymbol{x}}$当作被卷积对象，通过卷积操作计算卷积核相对于被卷积对象的相似度，完成相关运算。尽管这种相关运算方式的计算效率很高，但是由于计算过程中不涉及可学习参数，因此不能从数据中自适应地进行学习。

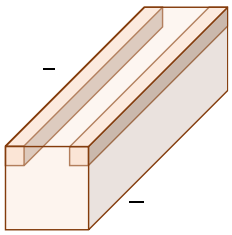

图 3　基于卷积神经网络的相关运算——直接卷积

(2)不对称卷积

如图 4 所示，针对直接卷积无可学习参数的问题，文献[14]提出一种采用不对称卷积（Asymmetric Convolution）进行相关运算的方法。

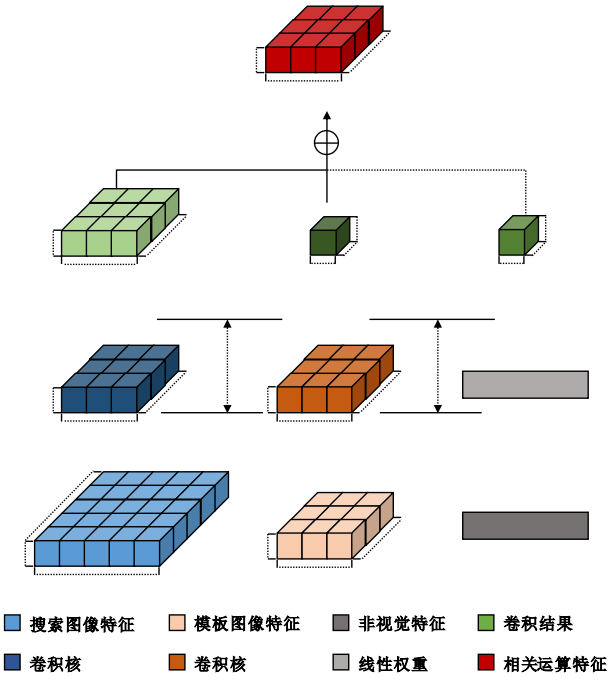

图 4　基于卷积神经网络的相关运算——不对称卷积[14]

该方法使用两个独立的、参数可学习的卷积核分别对搜索图像特征与模板图像特征进行卷积操作，如式 1.1 所示：

$$\boldsymbol{v}_i = [\boldsymbol{\theta}_z \quad \boldsymbol{\theta}_x] * \begin{bmatrix} \bar{\boldsymbol{z}} \\ \bar{\boldsymbol{x}}_i \end{bmatrix} = \boldsymbol{\theta}_z * \bar{\boldsymbol{z}} + \boldsymbol{\theta}_x * \bar{\boldsymbol{x}}_i \tag{1.1}$$

其中，$\bar{\boldsymbol{x}}_i \in \mathbb{R}^{C\times H_z\times W_z}$是搜索图像特征中按照模板图像特征尺寸分块得到的第$i$块，具有和模板图像特征一样的尺寸；$\boldsymbol{\theta}_z, \boldsymbol{\theta}_x \in \mathbb{R}^{P\times C\times H_z\times W_z}$是两个卷积核，用于分别对模板图像特征以及第$i$块搜索图像特征进行卷积操作；$\boldsymbol{v}_i \in \mathbb{R}^{P\times 1\times 1}$表示模板图像特征与第$i$块搜索图像特征进行相关运算的结果。

当下标i遍历所有n块搜索图像特征子区域的时候，便可以获得$\bar{\boldsymbol{z}}$和$\bar{\boldsymbol{x}}$的相关运算结果：

$$\boldsymbol{v} = \{\boldsymbol{v_i} | i \in [1,n]\} = \{\boldsymbol{\theta}_z * \bar{\boldsymbol{z}} + \boldsymbol{\theta}_x * \bar{\boldsymbol{x}}_i | i \in [1,n]\} = \boldsymbol{\theta}_z * \bar{\boldsymbol{z}} +_b \boldsymbol{\theta}_x * \bar{\boldsymbol{x}} \tag{1.2}$$

其中$+_b$是广播求和机制。

若存在手工设计的非视觉特征，则可以很方便地采用可学习的线性权重进行特征变换，得到的结果仍可以按照广播求和机制与$\boldsymbol{\theta}_x$和$\bar{\boldsymbol{x}}$的卷积结果相加。

不对称卷积可以直接将搜索图像所有子区域的特征当成一个整体来处理，这样可以以较低的计算量实现相关运算的离线可学习。但是这种相关运算方式只在模板图像特征和搜索图像特征的空间维度进行相关运算，没有考虑到特征的通道维度的相关性，因此难以激活更多的特征通道。

(3)“空间-通道”卷积

为了能够将特征同时在空间维度和通道维度进行相关运算，文献[15]将模板图像特征$\bar{\boldsymbol{x}}$分解为尺寸为1×1的空间核和通道核来分别与搜索图像特征进行相关运算，该工作指出这种方式可以减少匹配区域，同时抑制背景干扰且准确采集目标区域上的响应点。

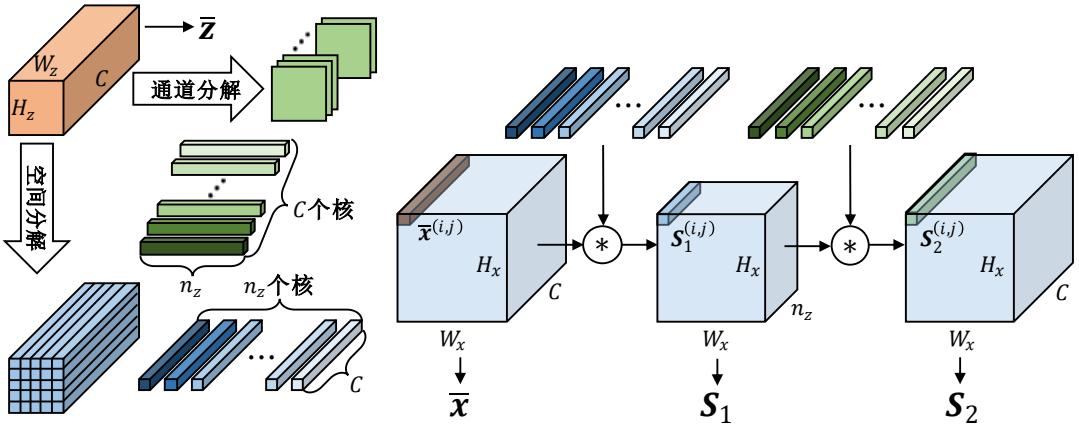

图 5　基于卷积神经网络的相关运算——“空间-通道”卷积[15]

如图 5 左侧所示，模板图像特征$\bar{\boldsymbol{z}}$在空间维度沿着长和宽被切分为n_z份，从而得到n_z个长度为模板图像特征通道数c的空间核：

$$\bar{\boldsymbol{z}}_S = \{\bar{\boldsymbol{z}}_S^1, \bar{\boldsymbol{z}}_S^2, \cdots, \bar{\boldsymbol{z}}_S^{n_z}\}$$

其中$n_z = W_z \times H_z$，W_z和H_z分别是模板图像特征的长和宽。

同时，为了在通道维度进行相关运算，再将模板图像特征$\bar{\boldsymbol{z}}$在通道维度切成c份，每份是一张$W_z \times H_z$的模板图像特征，拉直成为通道核：

$$\bar{\boldsymbol{z}}_C = \{\bar{\boldsymbol{z}}_C^1, \bar{\boldsymbol{z}}_C^2, \cdots, \bar{\boldsymbol{z}}_C^c\}$$

其中每个通道核的尺寸是$1 \times 1 \times n_z$。

相关运算的计算方式如图 5 右侧所示，其中W_x和H_x分别是搜索图像特征$\bar{\boldsymbol{x}}$的长和宽。记搜索图像特征的第i行、第j列为$\bar{\boldsymbol{x}}^{(i,j)}$，首先计算和空间核的相关性：

$$S_1^{(i,j)}[m] = \bar{\boldsymbol{x}}^{(i,j)} \cdot \bar{\boldsymbol{z}}_C^m \qquad m = 1,2,\cdots,n_z \tag{1.3}$$

其中$S_1^{(i,j)}$代表$\bar{\boldsymbol{x}}^{(i,j)}$和$\bar{\boldsymbol{z}}$在空间维度上第$m$个位置的相关性。

再计算和通道核的相关性：

$$S_2^{(i,j)}[n] = S_1^{(i,j)} \cdot \bar{\boldsymbol{z}}_C^n \qquad n = 1,2,\cdots,c \tag{1.4}$$

最后得到和搜索图像特征$\bar{\boldsymbol{x}}$尺寸相同的结果S_2，然后将S_2和$\bar{\boldsymbol{x}}$按通道拼接并用 1×1 卷积降维，最终得到相关运算的结果。

基于“空间-通道”卷积的相关运算把模板图像特征变成1×1的核，每次匹配的区域也是1×1大小，其中包含的背景信息便大大减少。空间核关注模板每个区域的信息，而通道核关注模板整体的信息，从而实现高效深度地融合空间特征和通道特征。

1.2.2 基于图神经网络的相关运算技术

如图 6 所示，文献[16]将图结构引入相关运算之中，其特点是图结构的引入将会构建模板图像特征与搜索图像特征的拓扑关系，相关运算则是在这种拓扑关系上进行的。

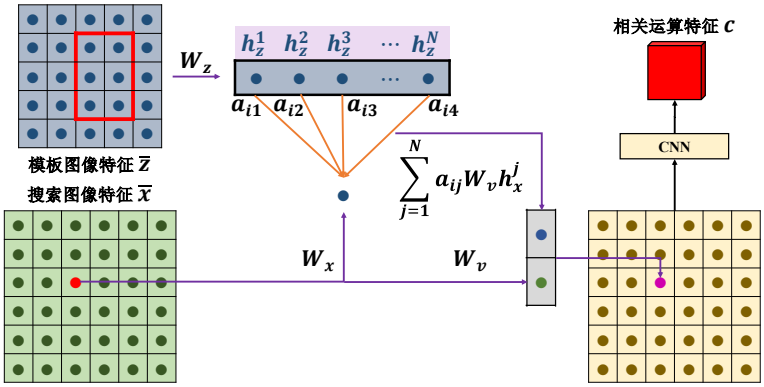

图 6　基于图神经网络的相关运算[16]

该工作将模板图像特征中的像素点和搜索图像特征中的像素点看成是一张二部图的两类节点，并在两类节点间构建连接，形成二部图的边。

模板图像特征$\bar{\boldsymbol{z}}$和搜索图像特征$\bar{\boldsymbol{x}}$的每一个$1\times1\times C$的像素点看成一个节点，其中C代表特征通道数。令V_z是特征$\bar{\boldsymbol{z}}$上所有节点的集合，V_x是特征$\bar{\boldsymbol{x}}$上所有节点的集合。使用一个完全二分图$G=(V,E)$来构建模板图像特征与搜索图像特征之间的对应关系，其中$V=V_z\cup V_x$，$E=\{(u,v)|\forall u\in V_x,\forall v\in V_z\}$。进一步定义$G$的两个子图$G_z=(V_z,\emptyset)$，$G_x=(V_x,\emptyset)$，其中$\emptyset$指空集；

采用图注意力机制[17]对模板图像特征与搜索图像特征进行相关运算。

对于每一个$(i,j)\in E$，令$\boldsymbol{e}_{ij}$表示节点$i\in V_x$和节点$j\in V_z$之间的相关得分：

$$\boldsymbol{e}_{ij}=f\left(\boldsymbol{h}_x^i,\boldsymbol{h}_z^j\right) \tag{1.5}$$

其中，$\boldsymbol{h}_x^i,\boldsymbol{h}_z^j\in\mathbb{R}^{C\times1\times1}$分别是节点$i$和节点$j$的特征向量。因为搜索图像特征中的某个位置与模板图像特征的局部特征越相似，它就越有可能是前景位置，所以需要将更多的目标信息从模板图像特征传递到该位置。基于上述考虑，将得分$\boldsymbol{e}_{ij}$设置成正比于两个节点特征之间的相似性。

使用特征之间的内积作为相似性度量。为了自适应学习节点之间更好的表示，先对节点特征进行线性变换，然后在变换后的特征向量之间取内积来计算相关得分：

$$f\left(\boldsymbol{h}_x^i,\boldsymbol{h}_z^j\right)=\left(\boldsymbol{W}_x\boldsymbol{h}_x^i\right)^T\left(\boldsymbol{W}_z\boldsymbol{h}_z^j\right) \tag{1.6}$$

其中$\boldsymbol{W}_x$和$\boldsymbol{W}_z$是线性变换矩阵。

为了平衡发送到搜索区域的信息量，使用 Softmax 函数将$\boldsymbol{e}_{ij}$正则化：

$$\boldsymbol{a}_{ij}=\frac{exp\left(\boldsymbol{e}_{ij}\right)}{\sum_{k\in V_z}exp(\boldsymbol{e}_{ik})} \tag{1.7}$$

其中$\boldsymbol{a}_{ij}$从节点j的视角衡量了跟踪器向节点i投入多少注意力。利用从G_z中所有节点传递到G_x中的第i个节点的注意力，计算节点i的聚合表示：

$$\boldsymbol{v}_i=\sum_{j\in V_z}\boldsymbol{a}_{ij}\boldsymbol{W}_v\boldsymbol{h}_z^j \tag{1.8}$$

其中$\boldsymbol{W}_v$是线性变换矩阵，$\boldsymbol{a}_{ij}$是注意力权重。

最后将聚合的特征与节点特征$\boldsymbol{h}_x^i$融合，得到一个基于目标信息的更强大的特征

表示：

$$\widehat{\boldsymbol{h}}_x^i \quad ReLU\left(\boldsymbol{v}_i \,\|\, \left(\boldsymbol{W}_v \boldsymbol{h}_x^i\right)\right) \tag{1.9}$$

其中‖表示向量的拼接。

对于任意节点$i \in V_x$计算$\widehat{\boldsymbol{h}}_x^i$，最终完成相关运算。

该方法采用将模板图像特征中带有目标信息的部分与整个搜索图像特征建立二部图，再采用图注意力机制在特征的空间维度进行相关运算，从模板中引入了较少背景噪声提升了跟踪效果。然而，它过于依赖模板图像特征中带有目标信息的部分的先验，以及缺乏通道维度的特征融合，使其在跟踪未知目标并且目标发生大形变的时候效果较差。

1.2.3 基于 Transformer 的相关运算技术

文献[18]设计出了一种基于注意力机制的新型网络结构用以处理机器翻译等任务并命名为 Transformer，自此 Transformer 在处理一维序列数据上展现出强大的性能。文献[19]设计了能够处理二维图像数据的 Transformer，成功将其引入计算机视觉领域并在各类任务中取得优越的性能表现。

为了将 Transformer 结构引入孪生网络目标跟踪中的相关运算，文献[20]改进了文献[19]中的视觉 Transformer 结构，设计了基于自注意力机制的自主语境增强（Ego-Context Augment, ECA）模块用以更好的捕获模板图像特征$\bar{\boldsymbol{z}}$与模板图像特征$\boldsymbol{z}$之间的语义信息，以及搜索图像$\bar{\boldsymbol{x}}$与搜索图像$\boldsymbol{x}$之间的语义信息；基于互注意力机制的交叉特征增强（Cross-Feature Augment, CFA）模块可以更好地捕获模板图像特征$\bar{\boldsymbol{z}}$和搜索图像特征$\bar{\boldsymbol{x}}$之间的语义信息。

如图 7 所示，首先对模板图像特征$\bar{\boldsymbol{z}}$和搜索图像特征$\bar{\boldsymbol{x}}$使用 1×1 卷积进行降维，得到$\bar{\boldsymbol{z}}_0$和$\bar{\boldsymbol{x}}_0$，其中$\bar{\boldsymbol{z}}_0 \in \mathbb{R}^{d \times H_z \times W_z}$，$\bar{\boldsymbol{x}}_0 \in \mathbb{R}^{d \times H_x \times W_x}$。由于 Transformer 只能输入序列信息，因此将$\bar{\boldsymbol{z}}_0$和$\bar{\boldsymbol{x}}_0$沿着空间维度拉直成长条状的形符（Tokens），记为$\bar{\boldsymbol{z}}_1 \in \mathbb{R}^{d \times H_z W_z}$和$\bar{\boldsymbol{x}}_1 \in \mathbb{R}^{d \times H_x W_x}$。

其中$\bar{\boldsymbol{z}}_1$拆分成$\{\bar{\boldsymbol{z}}_1^i\}_{i=1,2,\cdots,H_z \times W_z}$，$\bar{\boldsymbol{x}}_1$拆分成$\{\bar{\boldsymbol{x}}_1^j\}_{j=1,2,\cdots,H_x \times W_x}$，$\bar{\boldsymbol{z}}_1^i, \bar{\boldsymbol{x}}_1^j \in \mathbb{R}^{d \times 1 \times 1}$。将得到的$\{\bar{\boldsymbol{z}}_1^i\}$和$\{\bar{\boldsymbol{x}}_1^j\}$输入到基于 Transformer 的相关运算模块进行计算。

同原始的 Transformer 的文献[18]一致，ECA 模块的运算和 CFA 模块的运算都需要查询（query）算子、键（key）算子和值（value）算子。分别采用三个线性层来产生：

$$\begin{cases} \boldsymbol{q}_z = \mathrm{Linear}_q(\bar{\boldsymbol{z}}_1), & \boldsymbol{q}_x \quad \mathrm{Linear}_q(\bar{\boldsymbol{x}}_1) \\ \boldsymbol{k}_z = \mathrm{Linear}_k(\bar{\boldsymbol{z}}_1), & \boldsymbol{k}_x \quad \mathrm{Linear}_k(\bar{\boldsymbol{x}}_1) \\ \boldsymbol{v}_z = \mathrm{Linear}_v(\bar{\boldsymbol{z}}_1), & \boldsymbol{v}_x \quad \mathrm{Linear}_v(\bar{\boldsymbol{x}}_1) \end{cases} \tag{1.10}$$

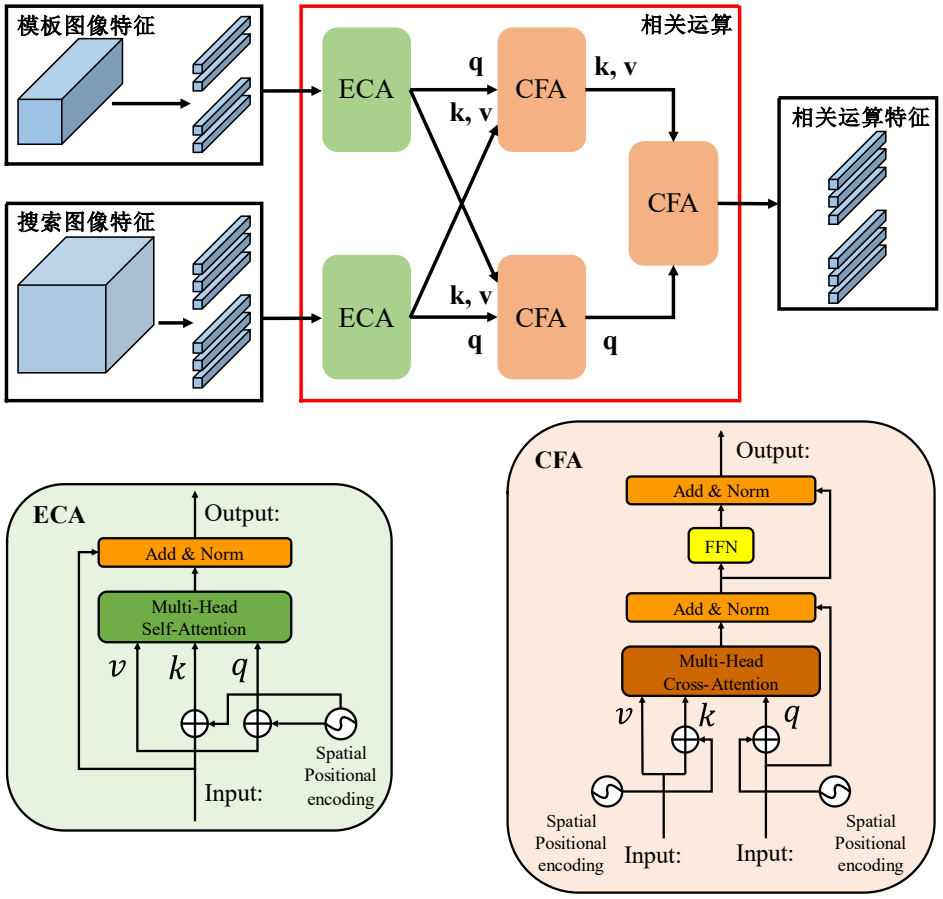

图 7　基于 Transformer 的相关运算[20]

ECA 中的自注意力机制如式 1.11：

$$\begin{cases} \mathrm{Self\_Attention}(\boldsymbol{q}_z, \boldsymbol{k}_z, \boldsymbol{v}_z) = \mathrm{softmax}\left(\dfrac{\boldsymbol{q}_z \boldsymbol{k}_z^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{v}_z \\ \mathrm{Self\_Attention}(\boldsymbol{q}_x, \boldsymbol{k}_x, \boldsymbol{v}_x) = \mathrm{softmax}\left(\dfrac{\boldsymbol{q}_x \boldsymbol{k}_x^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{v}_x \end{cases} \tag{1.11}$$

CFA 中的互注意力机制如式 1.12：

$$\begin{cases} \mathrm{Cross\_Attention}(\boldsymbol{q}_z, \boldsymbol{k}_x, \boldsymbol{v}_x) = \mathrm{softmax}\left(\dfrac{\boldsymbol{q}_z \boldsymbol{k}_x^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{v}_x \\ \mathrm{Cross\_Attention}(\boldsymbol{q}_x, \boldsymbol{k}_z, \boldsymbol{v}_z) = \mathrm{softmax}\left(\dfrac{\boldsymbol{q}_x \boldsymbol{k}_z^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{v}_z \end{cases} \tag{1.12}$$

其中d_k是缩放因子，一般设置为$\boldsymbol{k}$的维度。

在代码实现时往往与文献[18,19]中一致，采用多头注意力机制，它的计算如式 1.13 所示：

$$\begin{aligned}
\boldsymbol{o}_z &= \text{Multihead_Self_Attention}(\boldsymbol{q}_z,\boldsymbol{k}_z,\boldsymbol{v}_z) \quad \text{Concat}\left(\boldsymbol{H}_z^1,\cdots,\boldsymbol{H}_z^{n_h}\right)\boldsymbol{W}_z^O \\
\boldsymbol{o}_x &= \text{Multihead_Self_Attention}(\boldsymbol{q}_x,\boldsymbol{k}_x,\boldsymbol{v}_x) \quad \text{Concat}\left(\boldsymbol{H}_x^1,\cdots,\boldsymbol{H}_x^{n_h}\right)\boldsymbol{W}_x^O \\
\boldsymbol{o}_{zx} &= \text{Multihead_Cross_Attention}(\boldsymbol{q}_z,\boldsymbol{k}_x,\boldsymbol{v}_x) \quad \text{Concat}\left(\boldsymbol{H}_{zx}^1,\cdots,\boldsymbol{H}_{zx}^{n_h}\right)\boldsymbol{W}_{zx}^O \\
\boldsymbol{o}_{xz} &= \text{Multihead_Cross_Attention}(\boldsymbol{q}_x,\boldsymbol{k}_z,\boldsymbol{v}_z) \quad \text{Concat}\left(\boldsymbol{H}_{xz}^1,\cdots,\boldsymbol{H}_{xz}^{n_h}\right)\boldsymbol{W}_{xz}^O
\end{aligned} \tag{1.13}$$

其中：

$$\begin{aligned}
\boldsymbol{H}_z^i &\quad \text{Self_Attention}\left(\boldsymbol{q}_z\boldsymbol{W}_{\boldsymbol{q}_z}^i,\boldsymbol{k}_z\boldsymbol{W}_{\boldsymbol{k}_z}^i,\boldsymbol{v}_z\boldsymbol{W}_{\boldsymbol{v}_z}^i\right) \\
\boldsymbol{H}_x^i &\quad \text{Self_Attention}\left(\boldsymbol{q}_x\boldsymbol{W}_{\boldsymbol{q}_x}^i,\boldsymbol{k}_x\boldsymbol{W}_{\boldsymbol{k}_x}^i,\boldsymbol{v}_x\boldsymbol{W}_{\boldsymbol{v}_x}^i\right) \\
\boldsymbol{H}_{zx}^i &\quad \text{Cross_Attention}\left(\boldsymbol{q}_z\boldsymbol{W}_{\boldsymbol{q}_z}^i,\boldsymbol{k}_x\boldsymbol{W}_{\boldsymbol{k}_x}^i,\boldsymbol{v}_x\boldsymbol{W}_{\boldsymbol{v}_x}^i\right) \\
\boldsymbol{H}_{xz}^i &\quad \text{Cross_Attention}\left(\boldsymbol{q}_x\boldsymbol{W}_{\boldsymbol{q}_x}^i,\boldsymbol{k}_z\boldsymbol{W}_{\boldsymbol{k}_z}^i,\boldsymbol{v}_z\boldsymbol{W}_{\boldsymbol{v}_z}^i\right)
\end{aligned} \tag{1.14}$$

$\boldsymbol{W}_{z,x}^O,\boldsymbol{W}_{\boldsymbol{q}_{z,x}}^i\ \boldsymbol{W}_{\boldsymbol{k}_{z,x}}^i\ \boldsymbol{W}_{\boldsymbol{v}_{z,x}}^i\ \boldsymbol{W}_{zx}^O,\boldsymbol{W}_{xz}^O$都是参数矩阵，$n_h$表示多头注意力的头数。

在文献[20]设计的相关运算模块的第一层中，$\bar{\boldsymbol{z}}_1$自己的$\boldsymbol{q}_z,\boldsymbol{k}_z,\boldsymbol{v}_z$进行多头自注意力计算，即经过 ECA 模块处理；$\bar{\boldsymbol{x}}_1$自己的$\boldsymbol{q}_x,\boldsymbol{k}_x,\boldsymbol{v}_x$也进行多头自注意力计算，经过另一个 ECA 模块处理。经过 ECA 处理后的$\bar{\boldsymbol{z}}_1$的特征$\boldsymbol{o}_z$作为 CFA 模块的查询算子，经过 ECA 处理后的$\bar{\boldsymbol{x}}_1$的特征$\boldsymbol{o}_x$作为 CFA 模块的键算子和值算子，经过 CFA 模块后得到输出$\boldsymbol{o}_{zx}$；同理经过 ECA 处理后的$\bar{\boldsymbol{z}}_1$的特征$\boldsymbol{o}_z$作为另一个 CFA 模块的键算子和值算子，经过 ECA 处理后的$\bar{\boldsymbol{x}}_1$的特征$\boldsymbol{o}_x$作为另一个 CFA 模块的查询算子，经过该 CFA 模块后得到输出$\boldsymbol{o}_{xz}$。其余各层与第一层一致，文献[20]一共构建了 12 层这样的相关运算模块，最后输出模板图像特征和搜索图像特征深度融合的相关运算结果，实现更好地捕获语义信息的相关运算。然而，这样的语义信息往往是单一尺度的，不能很好地捕获形变目标的多尺度特征。

1.3 存在的难点

相关运算的本质是一种特殊的特征融合操作，需要将模板图像特征与搜索图像特征进行有效融合，以使用来自模板图像的目标信息指导搜索图像中的目标识别与定位。近年来，孪生网络目标跟踪中的相关运算成为一个热点问题，然而设计高效的相关运算目前仍然需要解决以下几个关键问题：

(1) 如何将图像特征在空间维度进行有效融合？

现有的相关运算技术大部分针对特征在空间维度的融合，主要使用基于卷积的相关运算和基于 Transformer 的相关运算。其中基于卷积的相关运算往往难以设计成像素级的相关运算，即模板图像特征中的每一个1×1的特征都分别与搜索图像特征中的每一个1×1的特征进行至少一次相关运算。图像卷积的特性决定了模板图像特征将以分块的形式与搜索图像特征整块进行相关运算，而像素级的相关运算需要将模板图像特征的分块细化到单个像素。因此对于基于卷积的相关运算，若不设计成像素级的相关运算，则会引入大量的背景噪声；若设计成像素级的相关运算，则会显著增加设计难度以及计算开销。此外，常见的卷积运算往往不带有注意力机制，而注意力机制已经被实验证明[19]对图像的特征提取是有意义的。基于 Transformer 的相关运算天然是一种像素级的相关运算，每一个 Token 即是一个1×1的特征，因此基于 Transformer 的相关运算自然具有很大的计算开销，而 Transformer 自身的结构设计加重了这种计算开销[19]。

因此，如何以较小的计算开销，在空间维度像素级地进行含注意力机制的相关运算有待进一步探究。

(2) 如何将图像特征在通道维度进行有效融合？

相关运算过程中，如果特征在通道维度较少地参与运算，则会导致有效激活的通道数较少，从而使得相关运算对目标边界的分辨能力较弱[14]，无法有效抵抗跟踪过程中出现的干扰物与目标形变。此外，特征在通道维度的相关性很难建模，目前缺乏一种有效的关系建模方式，能够在一定程度上适应通道维度抽象的潜在关系。通道维度往往不会设置过深，相关运算中模板图像特征和搜索图像特征常见的通道维度为 128 和 256，然而通道维度的相关运算需要遍历空间维度进行，因此通道维度的相关运算带来较大的计算开销仍然是一个问题。

因此，如何有效建模出特征通道维度的相关性，并以较小的计算开销，激活较多的通道维度参与相关运算有待进一步探究。

(3) 如何将图像的多尺度特征进行有效融合？

在直观上可以得到：让多个不同尺度的特征参与相关运算可以提高相关运算对多尺度特征处理的有效性，从而使得相关运算更好地适应目标的尺寸变化。尽管文献[12]中出现多尺度特征融合的思想，然而现有的相关运算技术几乎没有考虑多尺度特征的融合。

因此多尺度特征融合的必要性有待实验性的验证，如何设计高效融合多尺度特征的

模块有待进一步探究。

(4) 如何将设计的相关运算配适边缘设备的硬件环境进行实现？

随着相关运算研究的深入，各种复杂结构的相关运算设计层出不穷，然而有一些相关运算中涉及的算子很难在现有的边缘设备的硬件条件下进行实现，如地平线的旭日X3开发板并不直接支持图神经网络的相关算子、英伟达的 Jetson Nano 开发板难以支持 Transformer 模型的实时运行。

因此为了充分利用板上有限的硬件资源，设计出的相关运算必须进行优化。

1.4 论文的研究内容和贡献

本文针对孪生网络目标跟踪算法中相关运算的关键技术展开研究，研究目标、关键问题及研究内容和贡献如图 8 所示。

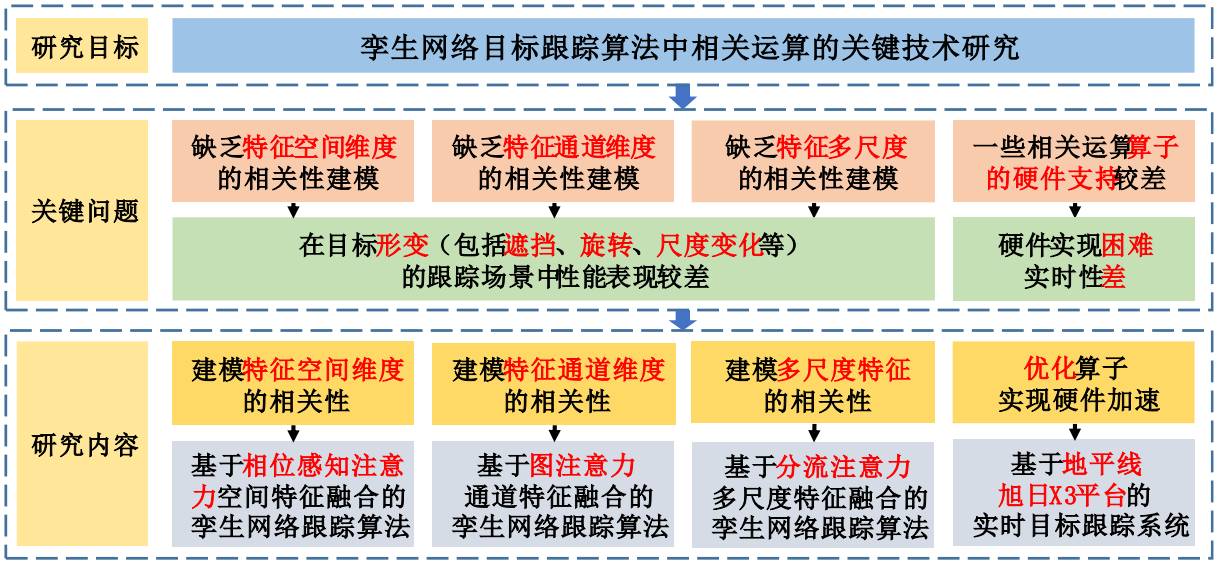

图 8　论文的主要研究内容和贡献

首先，空间维度的相关运算是最普遍的相关运算，本文首次提出一种基于多层感知机的相关运算，利用多层感知机这种最简单的结构学习特征如何在空间维度充分融合，针对空间维度的注意力机制，本文借用文献[21]的思想与结构，将特征当成“波”进行处理，并利用波的相位自适应地学习类似注意力机制的效果，实验表明这种采用简单多层感知机的相关运算相较于一般的基于卷积的相关运算表现更优。接着，通道维度的相关运算是常被忽略的相关运算，本文首次提出一种基于图神经网络的相关运算，借用文献[22]的思想，利用一种预先生成的图结构来建模通道维度潜在的相关性，并在这张图上采用图注意力卷积进行相关运算，实验表明这种方式相较于传统的在空间维度进行特

征融合的相关运算表现更优。最后，为了填补多尺度相关运算的空白，在不考虑计算开销的情况下，本文将多尺度特征融合的技术引入当前性能最好的基于 Transformer 的相关运算，借用文献[23]的思想，将注意力机制进行分流从而得到不同尺度的特征，并在不同尺度上进行相关运算，实验表明这种复杂的相关运算结构因为引入多尺度特征融合所以表现更优。

下面对本文的研究内容和相关贡献进行了总结：

(1) 基于相位感知注意力空间特征融合的孪生网络跟踪算法

为了解决现有的空间维度的像素级相关运算计算量较大且依赖注意力机制的问题，本文提出了一个基于相位感知注意力空间特征融合的孪生网络跟踪算法，该算法已经成功地应用于实际场景的视频单目标跟踪，并取得良好的效果。首先，模板图像特征和搜索图像特征会经过一个预融合模块，该模块以非常小的参数量实现了两种特征快速的、基本的相关运算，得到一个初步的相关运算结果。接着采用一种多层感知机结构，将初步的相关运算结果看作是“波”，像素级地分解其中的幅值和相位，不同的相位可以动态调节各像素块的权重，再采用欧拉公式进行展开得到实部和虚部，将相关运算视为一种实部和虚部的加权求和。实验表明算法在多个跟踪数据集上均展现了更高的跟踪性能与更快的处理速度。

(2) 基于图注意力通道特征融合的孪生网络跟踪算法

为了解决对通道维度特征之间潜在相关性的建模并依此进行相关运算，本文提出了一个基于图注意力通道特征融合的孪生网络跟踪算法，该算法已经成功地应用于实际场景的视频单目标跟踪，并取得良好的效果。首先，模板图像特征和搜索图像特征会经过一个预融合模块，该模块以非常小的参数量实现了两种特征快速的、基本的相关运算，得到一个初步的相关运算结果。接着采用一张预先生成的图来构建通道维度特征之间的关系，最后采用图注意力机制将通道维度特征按图结构进行融合。实验表明算法在多个跟踪数据集上均展现了更高的跟踪性能。

(3) 基于分流注意力多尺度特征融合的孪生网络跟踪算法

为了解决现有的相关运算技术缺乏对多尺度特征的融合，本文提出了一个基于分流注意力多尺度特征融合的孪生网络跟踪算法，该算法已经成功地应用于实际场景的视频单目标跟踪，并取得良好的效果。首先，模板图像特征和搜索图像特征会变换到合适维

度的特征空间，并被像素级地拆分为若干 Token。接着借鉴文献[20]中采用 Transformer 结构设计的 ECA 和 CFA 模块，但是使用分流注意力替换其中的注意力机制，设计用于相关运算的分流自注意力和分流互注意力，构建使用分流自注意力的 S-ECA 模块和使用分流互注意力的 S-CFA 模块。最后使用这两个模块构建实现多尺度特征融合的相关运算。实验表明算法在多个跟踪数据集上均展现了更高的跟踪性能。

(4) 基于地平线旭日 X3 平台的实时目标跟踪系统

本文结合论文所依托的项目，将上述跟踪算法针对地平线 X3 的硬件条件进行优化，最终应用在工程实践项目之中，且在项目的跟踪场景下展现了良好的跟踪性能，验证了本文算法的工程实用性。

1.5 论文的体系结构

本文针对孪生网络目标跟踪中的相关运算关键技术展开研究，通过从空间维度特征方面、通道维度特征方面、多尺度特征方面分析相关运算中的关键问题，分别提出了基于相位感知注意力空间特征融合的孪生网络跟踪算法、基于图注意力通道特征融合的孪生网络跟踪算法、基于分流注意力多尺度特征融合的孪生网络跟踪算法。最后，本文将所提出的算法应用在工程项目中，进一步验证算法的可行性。本文各章节之间的研究关系如图 9 所示。

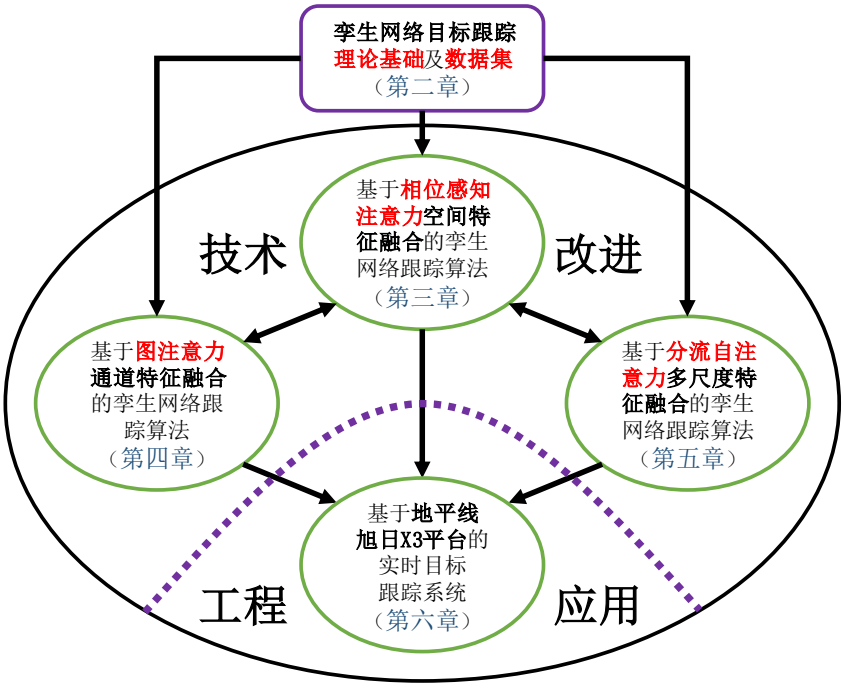

图 9　本文各章节之间的研究关系

第一章：绪论。首先针对孪生网络目标跟踪中的相关运算技术，介绍了相关的研究背景和意义。之后调研了相关运算的相关国内外研究现状，调研围绕着基于多层感知机的相关运算、基于卷积神经网络的相关运算、基于图神经网络的相关运算、基于Transformer 的相关运算四个方面展开。从空间维度特征、通道维度特征、多尺度特征三个方面总结目前该领域亟待解决的关键问题，并在此基础上介绍了本文的主要研究内容以及相关章节的组织安排。

第二章：基于相位感知注意力空间特征融合的孪生网络跟踪算法。空间维度特征的相关运算是最普遍的相关运算方式。先介绍所提出的预相关运算模块，再介绍所提出的基于多层感知机的相关运算，重点阐述其中利用波的相位实现类注意力机制的动态调节权重功能。本算法首次实现了使用简单的多层感知机完成相关运算，且相比一般的基于卷积的相关运算表现更优。

第三章：基于图注意力通道特征融合的孪生网络跟踪算法。阐述使用 Watts-Strogatz 算法生成随机图结构的原理，以及该图结构可以用以进行通道维度相关运算的原因。再使用图注意力机制进行图上的数据处理，实现通道维度特征的相关运算。本算法首次为相关运算建模了通道维度特征的相关性并进行相关运算，取得了比空间维度相关运算更好的效果。

第四章：基于分流注意力多尺度特征融合的孪生网络跟踪算法。阐述分流注意力的基本原理，并设计基于分流自注意力的自主语境增强模块、基于分流互注意力的交叉特征增强模块，并使用这两个模块完成相关运算，实现了多尺度特征的相关运算。在不考虑计算开销的情况下取得了极佳的跟踪性能。

第五章：基于地平线旭日 X3 平台的实时目标跟踪系统。前三章所设计的算法在边缘设备上进行实现需要进行大量的优化。针对地平线 X3 开发板，对相关运算进行一些配适板上硬件条件的优化，从而实现在边缘设备上的部署。部署在 X3 开发板上的跟踪算法被应用于飞机跟踪、导弹跟踪、车辆跟踪三个实际场景中的跟踪任务，完成相应的工程应用。

第二章　孪生网络目标跟踪理论基础

2.1 孪生网络目标跟踪流程

视频单目标跟踪的任务为：针对视频序列，首先根据初始视频帧中目标的状态，将跟踪器初始化，然后提取目标特征并建立目标模型，在后续视频帧中使用设计的跟踪策略，基于目标模型估计得到目标在当前帧的状态，最后利用当前状态更新目标模型，继续下一帧的跟踪。

基于孪生网络的单目标跟踪算法采用离线训练和在线应用的方式。先采用如图 2 的框架进行训练，再将训练得到的模型应用于在线的跟踪。

2.1.1 离线训练

如图 2 所示，常见的孪生网络目标跟踪算法离线训练的步骤分为五步：

①训练数据的准备：

整个孪生网络目标跟踪算法离线训练的输入，包括若干张模板图像和搜索图像。模板图像和搜索图像均从现存的跟踪数据集中抽取，以 LaSOT 数据集[24]为例：

如图 10 所示，上下两排图像序列分别从 LaSOT 数据集的 cat-13 视频序列文件夹和 rabbit-13 视频序列文件夹中抽取，序列图像中的目标分别是黑猫和兔子。

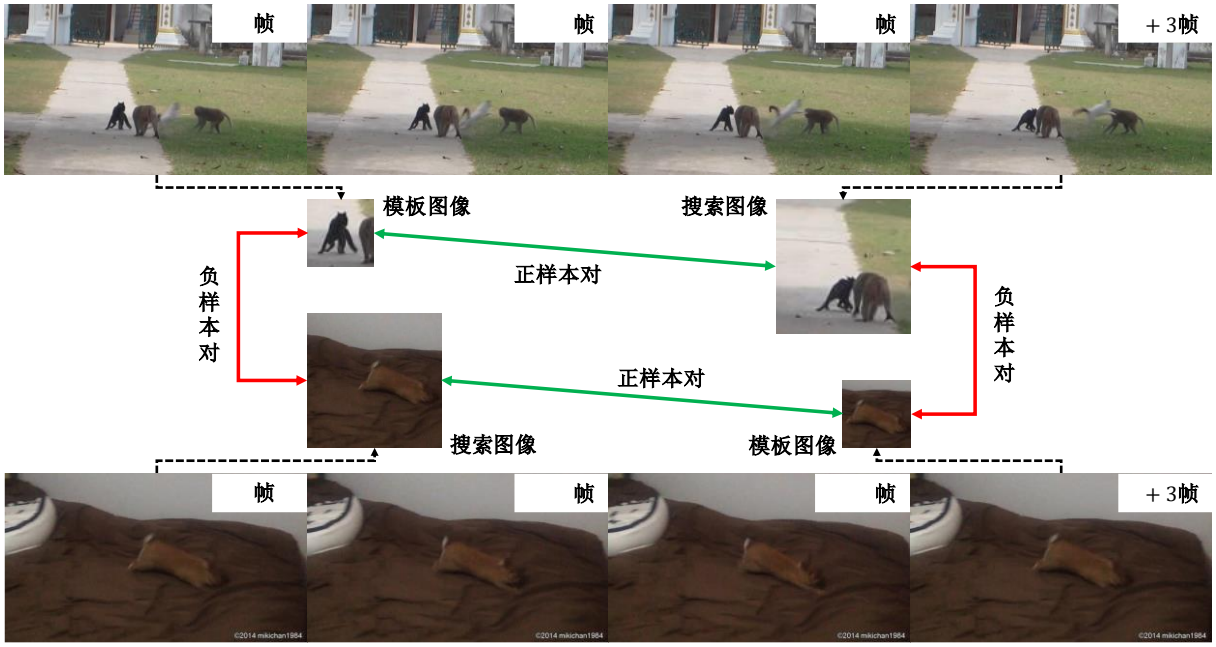

图 10　模板图像与搜索图像的抽取

抽取 cat-13 视频序列第t帧，在目标黑猫的周围裁剪一块正方形区域作为黑猫目标

的模板原始图像$\boldsymbol{z}$：

不妨设该帧图像标注的边界框（bounding box, bbox）为：

$$(x_0, y_0, x_1, y_1)$$

其中，x_0是 bbox 左上角的横坐标，y_0是 bbox 左上角的纵坐标，x_1是 bbox 右下角的横坐标，y_0是 bbox 右下角的纵坐标。

则该 bbox 的宽w和高h为：

$$\begin{cases} w = x_1 - x_0 \\ h = y_1 - y_0 \end{cases} \tag{2.1}$$

该 bbox 的中心坐标(x_c, y_c)为：

$$\begin{cases} x_c = (x_1 + x_0)/2 \\ y_c = (y_1 + y_0)/2 \end{cases} \tag{2.2}$$

一般裁剪的图片的边长r为：

$$r = \sqrt{[w + (w+h)/2] \cdot [h + (w+h)/2]} \tag{2.3}$$

记裁剪的 bbox 为(x_0', y_0', x_1', y_1')，则：

$$\begin{cases} x_0' = x_c - r/2 \\ y_0' = y_c - r/2 \\ x_1' = x_c + r/2 \\ y_1' = y_c + r/2 \end{cases} \tag{2.4}$$

此时可以按照裁剪的 bbox 裁剪得到黑猫目标的模板原始图像$\boldsymbol{z}$。黑猫目标的搜索原始图像$\boldsymbol{x}$可以同理裁剪得到，但是搜索原始图像的裁剪范围比模板原始图像更大，一般设置为模板原始图像的 2 倍尺寸。

图 10 表明，若黑猫目标的搜索原始图像$\boldsymbol{x}$从 cat-13 视频序列的其它帧裁剪得到（如第$t+3$帧）并且其中含有黑猫目标，则黑猫目标的模板原始图像$\boldsymbol{z}$和黑猫目标的搜索原始图像$\boldsymbol{x}$构成正样本对。若搜索原始图像$\boldsymbol{x}$从其它视频序列（如 rabbit-13）中随机抽取一帧并进行裁剪，则此时$\boldsymbol{z}$和$\boldsymbol{x}$构成负样本对，这是因为$\boldsymbol{x}$中并没有包含模板的黑猫目标。

本文后续涉及模板的变量均用字母 z 进行标记，涉及搜索的变量均用字母 x 进行标记。为了后续说明方便，记孪生网络目标跟踪算法离线训练的输入为一批（Batch）包含b张模板原始图像$\boldsymbol{z}$和b张搜索原始图像$\boldsymbol{x}$组成的样本对，包含的正负样本对的数量随机。

在下一步中骨干网络的输入尺寸一般在离线训练阶段固定，常用的设置为模板图像为 127×127 或者 128×128，搜索图像为 255×255 或者 256×256，因此需要将模板原始图像$\boldsymbol{z}$和搜索原始图像$\boldsymbol{x}$通过插值变换到该尺寸：

$$\begin{cases} \tilde{\boldsymbol{z}} = \text{Resize}(\boldsymbol{z}), & \tilde{\boldsymbol{z}} \in \mathbb{R}^{b\times\tilde{H}_z\times\tilde{W}_z}, \quad \tilde{H}_z = \tilde{W}_z = 127 \; or \; 128 \\ \tilde{\boldsymbol{x}} = \text{Resize}(\boldsymbol{x}), & \tilde{\boldsymbol{x}} \in \mathbb{R}^{b\times\tilde{H}_x\times\tilde{W}_x}, \quad \tilde{H}_x = \tilde{W}_x = 255 \; or \; 256 \end{cases} \tag{2.5}$$

其中，Resize为利用插值进行尺寸变换操作，$\tilde{\boldsymbol{z}}$为模板图像，$\tilde{\boldsymbol{x}}$为搜索图像。

②骨干（Backbone）部分进行特征提取：

孪生网络的“孪生”体现在分别采用两个结构一致且参数共享的骨干网络，同时对模板图像$\tilde{\boldsymbol{z}}$和搜索图像$\tilde{\boldsymbol{x}}$进行特征提取，记二者特征提取的结果为$\bar{\boldsymbol{z}}$和$\bar{\boldsymbol{x}}$，有：

$$\bar{\boldsymbol{z}} \in \mathbb{R}^{b\times c\times H_z\times W_z}, \bar{\boldsymbol{x}} \in \mathbb{R}^{b\times c\times H_x\times W_x}$$

其中，c是通道维度，常用设置为 256。搜索图像特征$\bar{\boldsymbol{x}}$的尺寸一般为搜索图像$\tilde{\boldsymbol{x}}$的尺寸的$1/8$，常用尺寸为 31 或者 32。

为了提取模板图像$\tilde{\boldsymbol{z}}$和搜索图像$\tilde{\boldsymbol{x}}$的深度特征，得到模板图像特征$\bar{\boldsymbol{z}}$和搜索图像特征$\bar{\boldsymbol{x}}$的高级表示，骨干网络一般采用在 ImageNet 数据集图像分类任务上取得良好性能表现的特征提取网络，包括但不限于多层感知机（Multi-Layer Perceptrons, MLP）网络、卷积神经网络（Convolution Neural Network, CNN）和视觉 Transformer 网络。

由于一些工作[11,12]已经深入探究了孪生网络目标跟踪中深度特征提取网络的设计，并且设计得到了相对固定的特征提取网络结构，因此本文将沿用这些工作的成果不再进行额外地探究。

③脖颈（Neck）部分进行特征融合：

早先的相关运算技术来源于图 3 所示的直接卷积，通过模板图像特征$\bar{\boldsymbol{z}}$在搜索图像特征$\bar{\boldsymbol{x}}$上匹配得到相关性，因此被称为 **Correlation 或 Match 操作**；在深度学习时代，研究发现可以使用带可学习参数的网络将$\bar{\boldsymbol{z}}$和$\bar{\boldsymbol{x}}$进行特征融合，通过数据自适应地学习两者之间的相关性，因此后来的相关运算技术又被称为 **Fusion 或 Integration 操作**。

从广义的角度来看，**相关运算技术的本质是一种特殊的特征融合运算。**

在 Neck 部分应用相关运算网络，计算模板图像特征$\bar{\boldsymbol{z}}$和搜索图像特征$\bar{\boldsymbol{x}}$的相关性，将模板图像特征$\bar{\boldsymbol{z}}$有效融合进搜索图像特征$\bar{\boldsymbol{x}}$，获得融合后的相关运算特征$\boldsymbol{o}$。

一般相关运算特征的尺寸与搜索图像特征$\bar{\boldsymbol{x}}$的尺寸保持一致：

$$\boldsymbol{o} \in \mathbb{R}^{b\times c'\times H_x\times W_x}$$

其中通道维度c'经常设置与c一致。

相关运算及相关运算网络将决定算法能否有效利用模板信息来对搜索图像中的目

标进行识别和定位，因此直接影响了跟踪算法的性能。

④头(Head)部分进行分类和回归：

一般采用若干卷积运算将相关运算特征$\boldsymbol{o}$变换为分类得分图$\boldsymbol{cls}$和回归得分图$\boldsymbol{reg}$。

其中：分类得分图$\boldsymbol{cls} \in \mathbb{R}^{b\times 2\times H_x\times W_x}$，用于目标识别（判断是否存在目标），第二维为 2 代表了存在目标（又称“前景”）和不存在目标（又称“背景”）两种类别。回归得分图$\boldsymbol{reg} \in \mathbb{R}^{b\times 4\times H_x\times W_x}$，用于目标定位（得到目标的边界框），第二维为 4 代表了锚点相对预测的 bbox 的左（Left）边界距离l，上（Top）边界距离t，右（Right）边界距离r，下（Bottom）边界距离b。

⑤损失函数的计算：

分类得分图$\boldsymbol{cls}$与回归得分图$\boldsymbol{reg}$对应的真实值（Ground Truth）如图 11 所示：

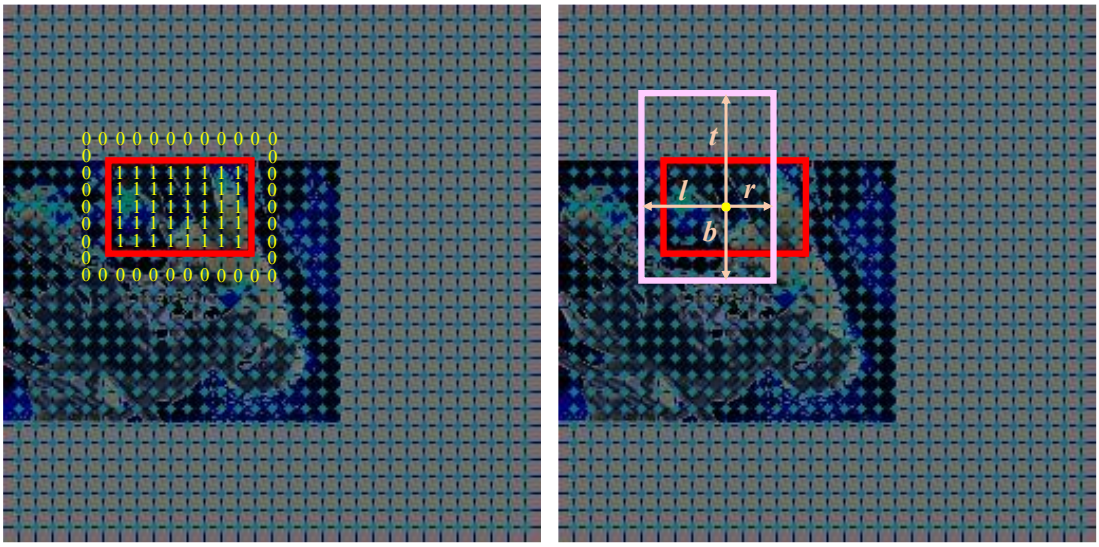

图 11　分类得分图（左）与回归得分图（右）

首先，搜索图像$\tilde{\boldsymbol{x}}$上将会以图像中心像素开始，向左、上、右、下每隔 stride 个像素标记一个锚点（Anchor），一个标记$H_x \times W_x$个锚点，其中 stride 的骨干网络的总步长。图 11 展示了一张 255×255 的搜索图像$\tilde{\boldsymbol{x}}$上被标记了 31×31 个浅蓝色的锚点，图中的 stride 取值为 8，搜索图像$\tilde{\boldsymbol{x}}$已经被一些增强操作进行了预处理。

分类得分图与回归得分图尺寸的$H_x \times W_x$与锚点数量对应，含义也对应有：

分类得分图$\boldsymbol{cls} \in \mathbb{R}^{b\times 2\times H_x\times W_x}$预测这$H_x \times W_x$个锚点在真实值对应的 bbox 内的概率，回归得分图$\boldsymbol{reg} \in \mathbb{R}^{b\times 4\times H_x\times W_x}$预测这$H_x \times W_x$个锚点相对于真实值对应的 bbox 的$l, t, r, b$的值。其中，真实值对应的 bbox 是图 11 中的红色边界框。

分类得分图$\boldsymbol{cls}$的真实值$\boldsymbol{cls}^*$如图 11 左图所示，真实值对应的 bbox 内的所有锚点分

类为 1，真实值对应的 bbox 外的所有锚点分类为 0。损失函数采用常见的二分类损失函数即可，如 BCE 损失函数：

$$\text{BCELoss}(\boldsymbol{cls}, \boldsymbol{cls}^*) = \frac{-1}{H_x \times W_x} \sum_i (\boldsymbol{cls}^*[i] * \log(\boldsymbol{cls}[i]) + (1 - \boldsymbol{cls}^*[i]) * \log(\mathbf{1} - \boldsymbol{cls}[i])) \tag{2.6}$$

其中，为了简便考虑略去批（Batch）数b，i遍历$H_x \times W_x$个锚点。

回归得分图$\boldsymbol{reg}$的真实值$\boldsymbol{reg}^*$如图 11 右图所示，真实值记为l^*, t^*, r^*, b^*，代表了全部$H_x \times W_x$个锚点相对于红色边界框的左边界距离、上边界距离、右边界距离和下边界距离。图 11 右图中粉色边界框是根据回归得分图$\boldsymbol{reg}$预测得到的l, t, r, b绘制的 bbox，当l, t, r, b越接近l^*, t^*, r^*, b^*时，粉色边界框与红色边界框的重叠程度越高。损失函数采用常见的 bbox 损失函数即可，如基于粉色边界框与红色边界框的交并比（Intersection over Union, IOU）的损失：

$$\text{IOULoss}(\boldsymbol{reg}, \boldsymbol{reg}^*) = -\log \frac{\text{Intersection}(l, t, r, b; l^*, t^*, r^*, b^*)}{\text{Union}(l, t, r, b; l^*, t^*, r^*, b^*)} \tag{2.7}$$

其中，Intersection函数可以利用l, t, r, b和l^*, t^*, r^*, b^*计算粉色边界框与红色边界框的交集的面积，Union函数可以利用l, t, r, b和l^*, t^*, r^*, b^*计算粉色边界框与红色边界框的并集的面积。

对上述的分类得分图与回归得分图的损失进行加权求和便可以得到总损失，再利用优化算法便可以进行损失对可学习参数的梯度的反向传播。

2.1.2 在线应用

图 12 展示了孪生网络目标跟踪算法在线应用时的流程。

将在目标跟踪数据集上训练好的网络应用于实际的在线跟踪遵循一定的步骤，下面以视频序列的前 3 帧为例说明：

①第一帧初始化跟踪器：

图 12 中黑色实线框内是 2.1.1 节训练好的网络，在本节中作为跟踪器使用。在线应用跟踪器对视频序列进行目标跟踪时，需要利用视频第一帧给出的目标的 ground truth 对跟踪器进行初始化，其中目标的 ground truth 是图 12 中第一帧图片中的红色 bbox。

如同离线训练一样可以根据第一帧图片中的红色 bbox 裁剪出模板原始图像$\boldsymbol{z}$，得到跟踪器的输入之一，即模板图像$\tilde{\boldsymbol{z}}$，经过骨干网络特征提取之后得到模板图像特征$\bar{\boldsymbol{z}}$，完

成跟踪器的初始化。

在线应用跟踪器进行目标跟踪时，一般令模板图像特征$\bar{\boldsymbol{z}}$在后续帧中保持不变，用以不断在搜索图像中识别与定位目标。同时，由于实际跟踪过程中的复杂情况，初始帧中得到的模板图像特征$\bar{\boldsymbol{z}}$可能不再适用于后续帧，一些工作探究了如何动态更新模板，但本文探究的内容不包括动态模板更新。

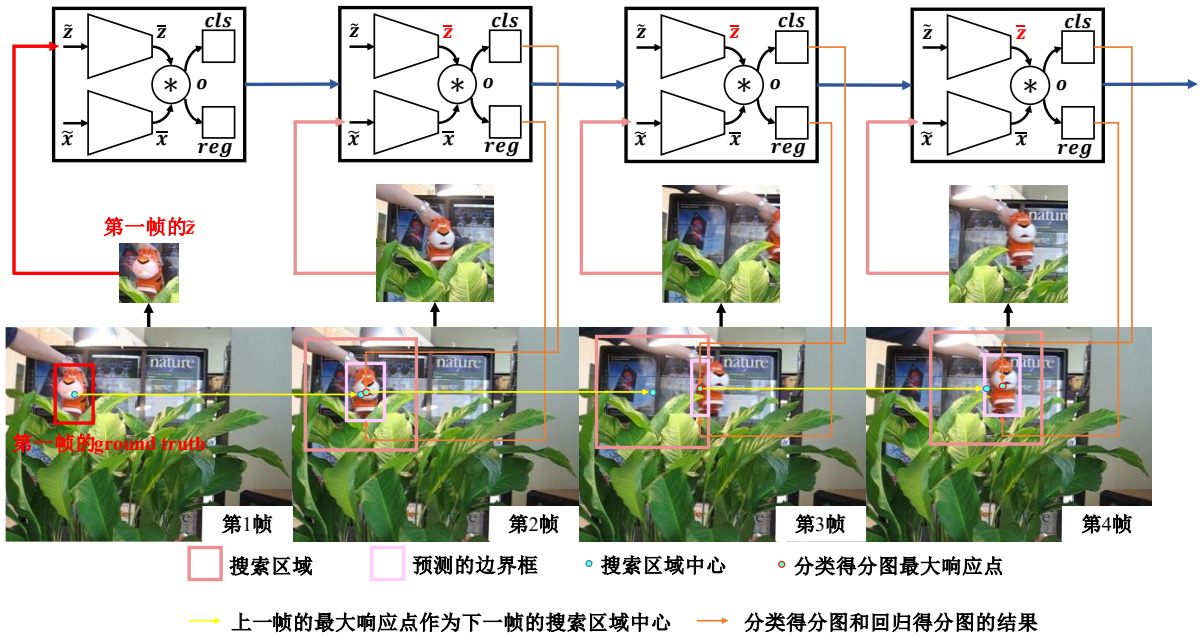

图 12　在线跟踪流程示意图

②第二帧开始跟踪：

记第一帧图像中的红色 bbox 的中心坐标为(u_0, v_0)，在第二帧中以(u_0, v_0)为中心裁剪出搜索区域，因此可以通过跟踪器的骨干网络得到搜索图像特征$\bar{x}$，进而可以得到分类得分图与回归得分图。选择分类得分图中得分最高的锚点，即第二帧图像中的最大响应点，以该点为中心绘制回归得分图预测的 bbox，此时便得到了第二帧图像中的目标的位置与边界框，完成了第二帧的跟踪。

③第三帧继续跟踪：

记第二帧图像中的最大响应点的坐标为(u_1, v_1)，在第三帧中以(u_1, v_1)为中心裁剪出搜索区域，其它操作同理第二帧可以完成第三帧的跟踪。

综上所述，后续帧中只要进行和第三帧同样的操作，便可以完成整个视频序列的目标跟踪。

2.2 数据集介绍

本节介绍本文工作中所使用的六种数据集，数据集中包含训练集与测试集，具体的使用方式见后续各章的实验部分。

2.2.1 GOT-10k

GOT-10k 是一个大规模的高多样性视频序列数据集[25]，它总共包含 9695 个视频序列，其中训练集 9335 个，验证集 180 个，测试集 180 个，总类别数为 563 类。数据集一共大约 66GB。GOT-10k 数据集于 2019 年由中科院自动化所发布，目前已经成为目标跟踪领域最常用的数据集之一，该数据集的示例图片如图 13 所示。

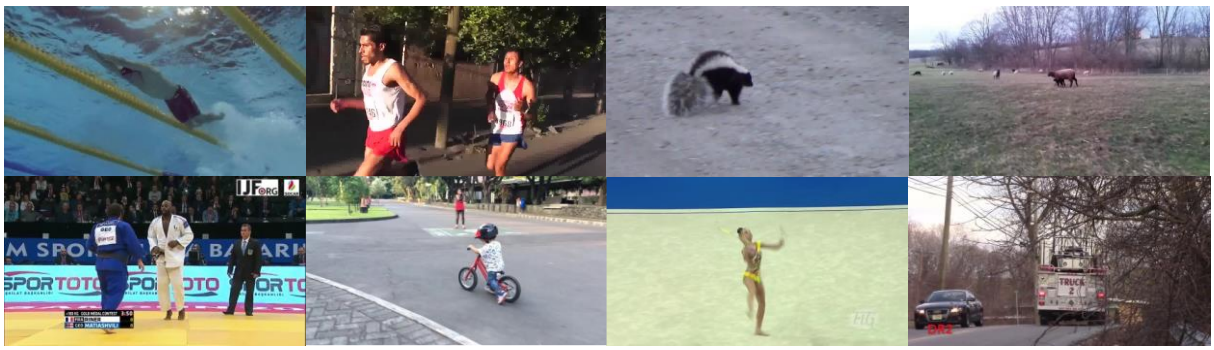

图 13　GOT-10k 数据集示例图片

该数据集的特点是规模大，具有近 10000 个视频序列和超过 150 万个标注框；标注的类别数多，高达 563 个类别；训练集和测试集的类别间没有重叠，使得评估结果更加鲁棒和泛化；此外还提供了除类别外的额外标签，如运动类型等。

2.2.2 YouTube-BoundingBoxes

YouTube-BoundingBoxes 是一种大型视频 URL 数据集[26]，简称 YT-BB。它总共包含大约 38000 个 15-20 秒的视频片段，总类别数为 23 类。YT-BB 数据集于 2017 年由谷歌公司发布，目前已经成为目标检测和跟踪领域常用的数据集之一。该数据集的示例图片如图 14 所示。

该数据集的特点是具有密集采样的人工注释，其分类标签和边界框（约 5.6 万个）精度高；视频中目标位移大，且视频未经过后期处理，包含遮挡、运动模糊和自然光照等因素，与真实应用场景高度一致；数据集为整个视频段提供边界框注释，对于遮挡等情况，可以在视频的上下文中理解、识别和跟踪。

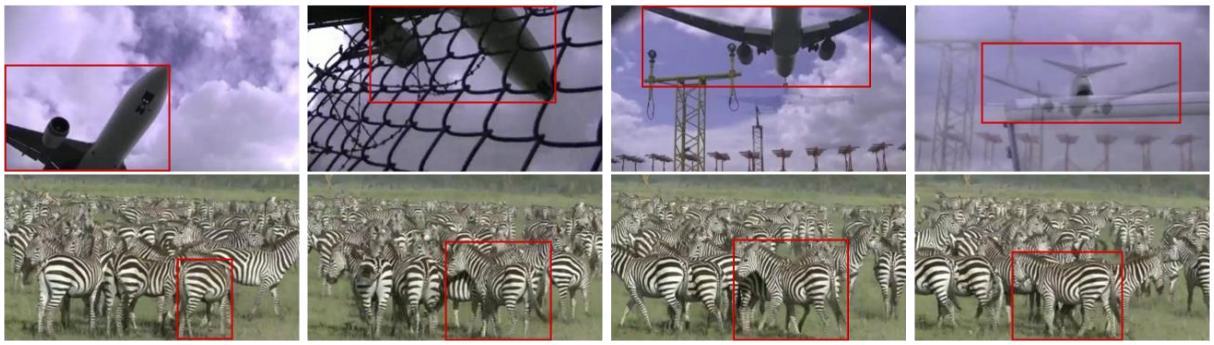

图 14　YT-BB 数据集示例图片

2.2.3 COCO

COCO 是一种大型图片数据集[27]，它总共包含大约 328000 张带有 2.5 万个标注的图片，总类别数为 91 类。COCO 数据集于 2014 年由微软公司首次发布，目前已经成为目标检测、语义分割和图像描述等领域最常用的数据集之一。该数据集的示例图片如图 15 所示。

图 15　COCO 数据集示例图片

该数据集的特点是规模大，具有近 330000 张图片和超过 20 万个标注；每张图片提供 5 个之多的丰富的描述；提供大约 1.5 万个实例对象的超像素级别的分割；此外还提供了额外标签，如人体关键点等。

2.2.4 DET/VID

ImageNet 是目前世界上最大的图像识别数据集[28]，它涵盖了 1000 个物体类别，包含了 1281167 张训练图像，50000 张验证图像和 100000 张测试图像。基于该数据集的挑战赛 ILSVRC（IMAGENET Large Scale Visual Recognition Challenge）主要分为五个场景：图像分类（CLS）、目标定位（LOC）、目标检测（DET）、视频目标检测（VID）和场景分类（Scene Classification）。ImageNet 数据集项目于 2009 年由斯坦福大学李飞

飞团队发起，该数据集的示例图片如图 16 所示。

在 ImageNet 的图像分类任务上有各种取得良好性能表现的网络结构，这些网络结构往往被进行微调（Fine-tune）并作为特征提取网络应用在其它任务（目标检测、目标分割等）；同样，这些结构也能启发孪生网络目标跟踪算法中的相关运算的设计。

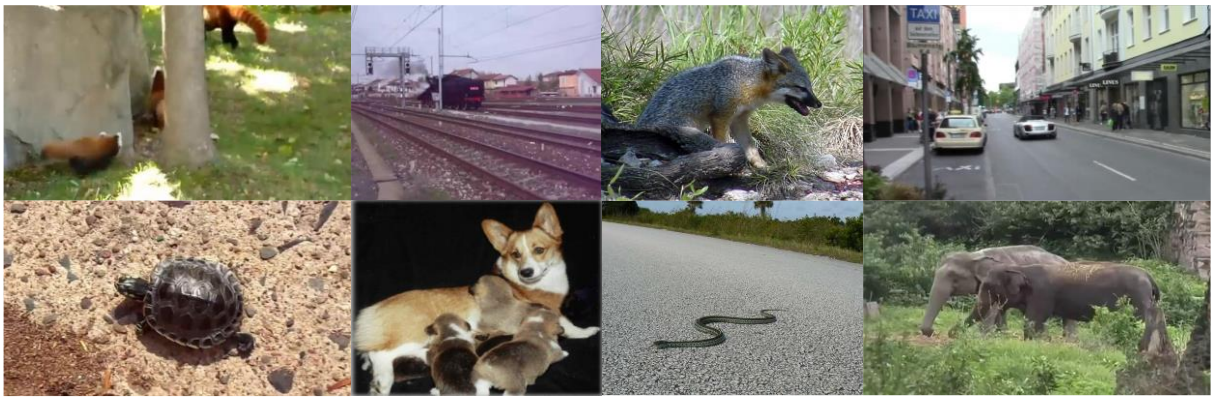

图 16　DET/VID 数据集示例图片

2.2.5 OTB100

OTB100 是一个用于评估跟踪算法性能的小型视频序列数据集[29]。它总共包含 100 个视频序列，总类别数为 22 类。全部视频序列用于跟踪算法的性能测试，因此无类别标注。数据集一共大约 2.7GB。OTB100 数据集于 2015 年由加州大学默塞德分校和汉阳大学联合发布，目前已经成为目标跟踪领域常用的测试数据集之一。该数据集的示例图片如图 17 所示。

图 17　DET/VID 数据集示例图片

该数据集的特点是包含了 25%的灰度数据和 75%的彩色数据；数据集被人工标注了除真实边界框外的属性，即涉及到光照变化、尺度变化、遮挡、目标形变、运动模糊、

快速运动、平面内旋转、平面外旋转、目标离开视野、背景干扰、低分辨率 11 个可能影响跟踪算法性能的属性。

2.2.6 LaSOT

LaSOT 是一个大型的单目标跟踪视频序列数据集[24]。它总共包含 1400 个视频序列，其中训练集 1120 个，测试集 280 个，总类别数为 70 类，每类 20 个视频。数据集平均每个视频 2512 帧，最短 1000 帧，最长 11397 帧。数据集一共大约 227GB。LaSOT 数据集于 2019 年由华南理工大学-鹏城实验室和天普大学等联合发布，目前已经成为目标跟踪领域最常用的数据集之一。该数据集的示例图片如图 18 所示。

图 18　DET/VID 数据集示例图片

该数据集的特点是规模大、类别多；数据集被密集标注且标注质量高；数据集还有丰富的自然语言描述；此外该数据集视频序列长度较长，可以被用来训练与评估长时跟踪算法。

2.3 评价指标与基线算法

2.3.1 平均重叠率与平均重叠率均值

平均重叠率（Average Overlap, AO）是 LaSOT 和 YT-BB 测试集中采用的评价指标。它计算跟踪算法生成的边界框与真实的边界框之间交并比（Intersection over Union, IOU）的平均值。

平均重叠率均值（mean Average Overlap, mAO）是一种总体性能评价标准。它对类别进行了归一化，如公式 2.8 所示：

$$\mathrm{mAO} = \frac{1}{C}\sum_{c=1}^{C}\left(\frac{1}{|S_c|}\sum_{i\in S_c}\mathrm{AO}_i\right) \tag{2.8}$$

其中C为类别数，S_c为属于第c类的序列子集。

2.3.2 成功率

成功率（Success Rate, SR）是 LaSOT 和 YT-BB 测试集中采用的评价指标。它统计 AO 超过一定阈值的视频帧数占视频所有帧数的百分比。

2.3.3 精确图与正则精确图

精确图（Precision Plot）是 OTB100 和 GOT-10k 测试集中采用的评价方法。它计算跟踪算法生成的边界框的中心位置与真实的边界框的中心位置直接的平均欧式距离，并称其为中心位置误差。精确图的横坐标为中心位置误差的阈值，纵坐标是中心位置误差小于阈值的视频帧数占视频所有帧数的百分比。

正则精确图（Normalized Precision Plot）考虑了真实框的尺度大小，在上述中心位置误差的基础上乘一个权重，该权重为真实框的对角线的距离。

2.3.4 成功率图

成功率图（Success Plot）是 OTB100 和 GOT-10k 测试集中采用的评价方法。它以跟踪算法生成的边界框和真实的边界框的 IOU 的阈值为横坐标，IOU 大于该阈值的视频帧数占视频所有帧数的百分比为纵坐标绘制图像。

SR_{50}和SR_{75}是成功率图的两种评价指标，分别表示阈值为 0.5 和 0.75 时 IOU 大于该阈值的视频帧数占视频所有帧数的百分比。该图还可以使用曲线下面积（Area Under Curve, AUC）指标进行评估，即计算 Success Plot 曲线下的面积。

2.3.5 基线算法

基线（Baseline）算法是各种算法性能比较中的“基准线”，一个算法以基线算法为“参照物”，便可以得到自身算法性能的相对好坏程度。

在本文后续各章算法的实验中选取 SiamDW[11]作为基线算法，该算法属于基于孪生网络的单目标跟踪算法，其中的骨干部分已经替换成常用的 ResNet50[40] 特征提取网络，脖颈部分的相关运算采用图 3 中的直接卷积。

为了与基线算法进行比较，本文后续章节中设计的孪生网络目标跟踪算法骨干部分与 SiamDW 保持一致，均设置为 ResNet50 特征提取网络；脖颈部分的相关运算被从空间特征融合、通道特征融合和多尺度特征融合三个方面进行改进，并与 SiamDW 以及其

它近年常见的优秀跟踪算法进行比较。

本文第三章、第四章、第五章、第六章的实验部分将会通过列表与绘图的方式详细展示并分析与基线算法的比较，得到本文设计的算法性能的相对好坏程度，证明设计的算法的有效性。

本文所有的算法及测试集上的主要指标的比较结果详见第六章的表 17。

第三章 基于相位感知注意力空间特征融合的孪生网络跟踪算法

特征在空间维度进行相关运算是主流的相关运算方式。然而现有的算法采用基于卷积或者基于 Transformer 的方式从而产生巨大的计算开销，且严重依赖传统的注意力机制。针对这些问题，本章提出了一种基于相位感知注意力的空间特征融合算法，可以在使用简单的多层感知机的情况下完成相关运算，并且以一种“波”的视角实现类注意力机制。

3.1 引言

当前计算机视觉领域主要研究重点是卷积神经网络和视觉 Transformer 网络，基于这两种网络设计的特征提取网络在图像分类任务上已经取得了丰富的成果。然而，受限于感知机的简单结构，简单地堆叠多个感知机往往不能有效地对图像数据进行特征提取，而且调参困难，容易陷入过拟合。

然而，最近的研究[21,30-37]表明，一些多层感知机（Multi-Layer Perceptrons, MLP）构成的网络可以以较小的参数量表现出比卷积神经网络更优的性能。表 1 列举了近些年采用多层感知机结构设计的网络及其在 ImageNet 数据集上完成图像分类任务的相关性能，其分类准确率指标已经超过了很多传统的深度卷积神经网络，如表 1 的结尾处列出的深度卷积神经网络中具有代表性的 ResNet[40]系列的网络及其性能指标。

表 1　近年视觉 MLP 网络在 ImageNet 分类任务上的表现

模型	参数量	FLOPs	Top-1 acc. (%)
EAMLP-14[30]	30M	-	78.9
EAMLP-19[30]	55M	-	79.4
Mixer-B/16[31]	59M	12.7G	76.4
ResMLP-S12[32]	15M	3.0G	76.6
ResMLP-S24[32]	30M	6.0G	79.4
ResMLP-B24[32]	116M	23.0G	81.0
gMLP-S[33]	20M	4.5G	79.6
gMLP-B[33]	73M	15.8G	81.6
S2-MLP-wide[34]	71M	14.0G	80.0
S2-MLP-deep[34]	51M	10.5G	80.7
ViP-Small/7[35]	25M	6.9G	81.5

ViP-Medium/7[35]	55M	16.3G	82.7
ViP-Large/7[35]	88M	24.4G	83.2
AS-MLP-T[36]	28M	4.4G	81.3
AS-MLP-S[36]	50M	8.5G	83.1
AS-MLP-B[36]	88M	15.2G	83.3
CycleMLP-B1[37]	15M	2.1G	78.9
CycleMLP-B2[37]	27M	3.9G	81.6
CycleMLP-B3[37]	38M	6.9G	82.4
CycleMLP-B4[37]	52M	10.1G	83.0
CycleMLP-B5[37]	76M	12.3G	83.2
Wave-MLP-T[21]	17M	2.4G	80.6
Wave-MLP-S[21]	30M	4.5G	82.6
Wave-MLP-M[21]	44M	7.9G	83.4
Wave-MLP-B[21]	63M	10.2G	83.6
ResNet18[40]	12M	1.8G	69.8
ResNet50[40]	26M	4.1G	78.5
ResNet101[40]	45M	7.9G	79.8

图 19 是一个由一层输入层与若干层隐藏层组成的多层感知机。可以看出，每一个圆圈代表一个神经元，每一层的神经元之间没有连接，每一层的神经元和下一层的神经元之间都有连接，因此被称为全连接层。隐藏层中的每一层的神经元接收上一层的各个神经元的输入后进行加权求和，其中的权重即为待训练参数。为了增加多层感知机的非线性，求和后的结果将会经过激活函数（Activation Function）。在训练过程中为了防止过拟合并加速训练，常常让一些神经元临时停止工作，这种技巧叫做丢弃（Dropout）。

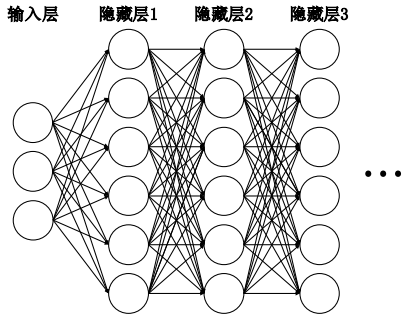

图 19　多层感知机结构

如图 20 所示，现有的基于多层感知机的网络结构的输入为图像块（image patches）或者图像形符（tokens），并且在空间维度设计 token-mixing MLP 模块进行融合，在通

道维度设计 channel-mixing MLP 进行融合。前者将不同 tokens 之间的信息进行融合，后者将同一 token 之间的各特征通道进行融合。

记一图像的特征张量$\boldsymbol{f} \in \mathbb{R}^{c\times H\times W}$，则特征张量$\boldsymbol{f}$在通道维度可以拆分为$c$个 token：

$$\boldsymbol{f}_C = \{\boldsymbol{f}_C^1, \boldsymbol{f}_C^2, \cdots, \boldsymbol{f}_C^c\}, \quad \boldsymbol{f}_C^i \in \mathbb{R}^{(W\times H)}, \quad i = 1,2,\cdots,c$$

同理特征张量$\boldsymbol{f}$在空间维度可以拆分为n个 token：

$$\boldsymbol{f}_S = \{\boldsymbol{f}_S^1, \boldsymbol{f}_S^2, \cdots, \boldsymbol{f}_S^n\}, \quad \boldsymbol{f}_S^j \in \mathbb{R}^c, \quad j = 1,2,\cdots,n$$

其中$n = W \times H$，W 和H分别是图像的特征张量的长和宽。

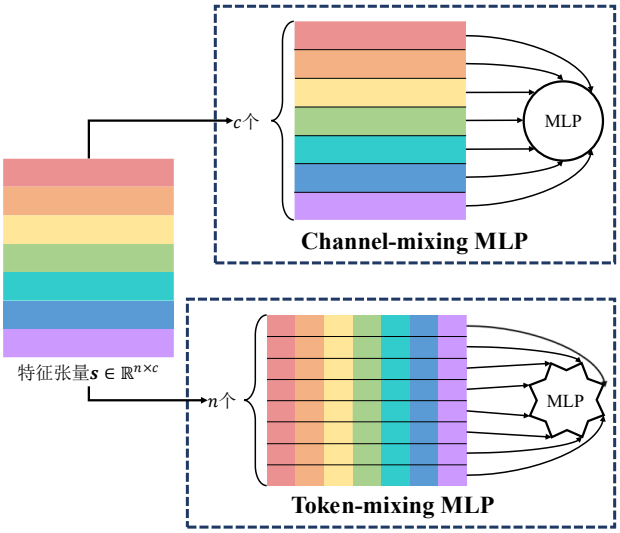

图 20　通道融合的多层感知机与空间融合的多层感知机

则图像的特征张量的 channel-mixing MLP 为：

$$\text{Channel\_MLP}\left(\boldsymbol{f}_C^i, \boldsymbol{W}^c\right) = \boldsymbol{W}^c \boldsymbol{f}_C^i, \qquad i = 1,2,\cdots,n \tag{3.1}$$

同理图像的特征张量的 token-mixing MLP 为：

$$\text{Token\_MLP}(\boldsymbol{f}_S, \boldsymbol{W}^t)_j = \sum\nolimits_k W_{jk}^t \odot \boldsymbol{f}_S^k, \qquad j = 1,2,\cdots,n \tag{3.2}$$

其中⊙为元素乘（Element-wise Multiplication）操作。

由式 3.1 和 3.2 可以看出，由多层感知机构成的网络可以仅由全连接层或者线性层进行堆叠实现，但是这样的网络结构在进行视觉任务的训练过程中也往往面临过拟合的风险，因此搭建的模型常常使用激活函数、批正则化（Batch Normalization）和丢弃操作。

然而，直接将多层感知机应用于相关运算将会面临一些问题。现有的基于多层感知机的网络的设计经验往往是针对图像分类等任务，需要处理的对象是单张图像或者是单个图像特征张量。在孪生网络目标跟踪的相关运算里，往往需要处理模板图像特征和搜

索图像特征两种图像特征张量，并需要将其有效融合。此外，注意力机制在这种融合中被证明是有必要的，这就要求设计的多层感知机网络也能应用注意力机制，同时需要考虑整个设计的相关运算具有可以接受的计算量。

为了解决上述问题，本章目标是基于多层感知机结构设计出一个能够有效融合模板图像特征和搜索图像特征两种图像特征张量的相关运算。首先，采用一些简单操作将模板图像特征和搜索图像特征进行预融合，设计预相关运算网络，得到两种特征初步融合的结果。然后介绍借用“波”的思想的多层感知机结构，并证明这种结构能够产生一种叫相位感知注意力机制的类注意力机制效果，即在不显式生成查询算子、键算子和值算子的情况下能够自主动态地调节各 tokens 之间地权重。最后，设计的基于相位感知注意力空间特征融合的孪生网络跟踪算法将在 GOT-10k、YT-BB、COCO、DET 和 VID 五个数据集的训练集上进行训练，在 OTB100、GOT-10k 和 LaSOT 三个数据集的测试集上进行测试。

3.2 预相关运算模块

预相关运算模块的基本处理流程如图 21 所示，包含三个关键步骤：像素级加和[38]（Point-wise Addition）、相关性匹配[38]（Pair-wise Relation）、仿射变换[38]（FiLM）。其中像素级加和模块中使用了文献[39]中的 RoIAlign 操作。

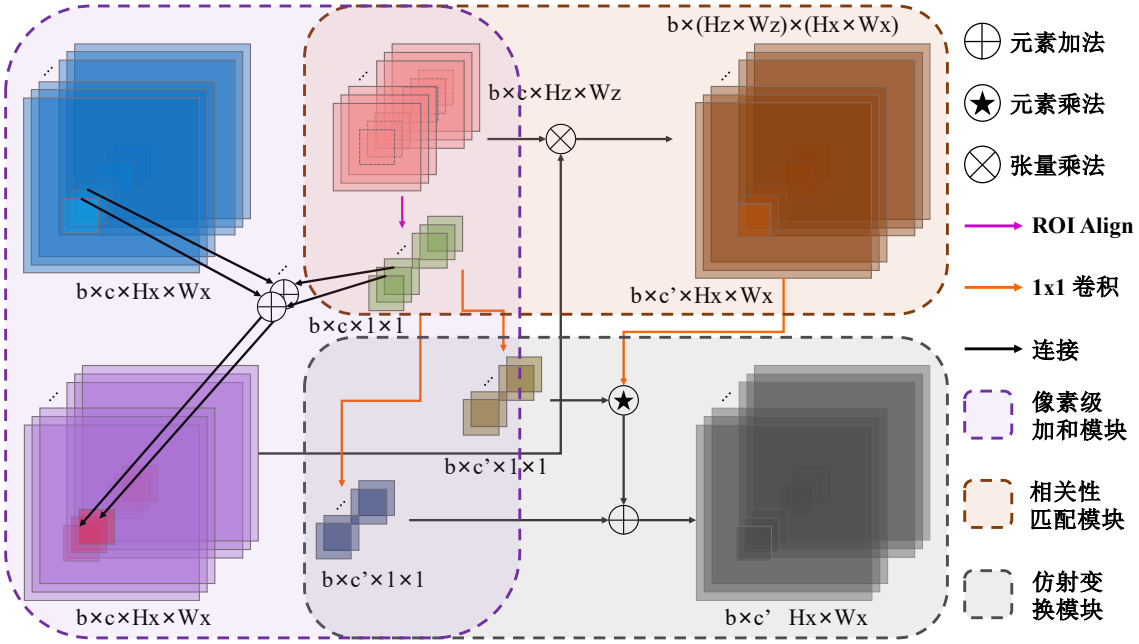

图 21　预相关运算模块

3.2.1 像素级加和操作

像素级加和操作的作用是将模板图像特征$\bar{\boldsymbol{z}}$上的感兴趣区域进行 RoIAlign 操作，并将得到的 RoIAlign 特征$\boldsymbol{r}$采用广播加和机制与搜索图像特征$\bar{\boldsymbol{x}}$进行求和。

图 22 展示了模板图像$\boldsymbol{z}$上目标的边界框与模板图像特征$\bar{\boldsymbol{z}}$上的感兴趣区域之间的关系，RoIAlign 操作基于此关系进行。

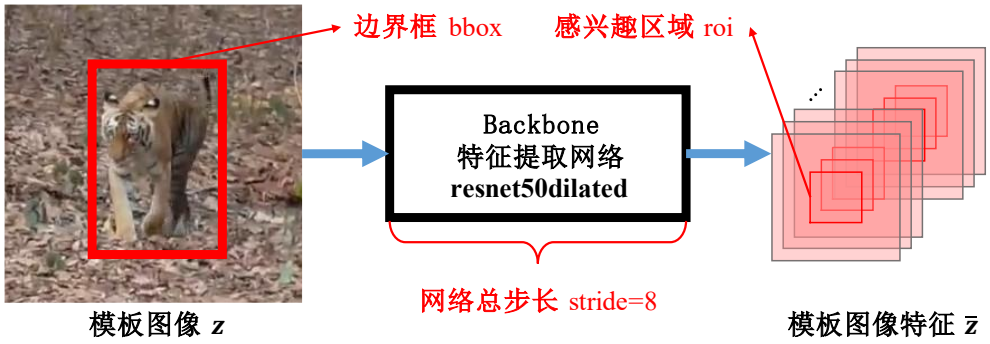

图 22 边界框与感兴趣区域的对应关系

设模板图像特征$\bar{\boldsymbol{z}} \in \mathbb{R}^{b\times c\times H_z\times W_z}$和搜索图像特征$\bar{\boldsymbol{x}} \in \mathbb{R}^{b\times c\times H_x\times W_x}$，则如图 23 所示：

$$\boldsymbol{r} = \mathrm{Conv}\big(\mathrm{RoIAlign}(\bar{\boldsymbol{z}}, bbox, stride)\big) \tag{3.3}$$

其中，b是单批（Batch）图像的数量，$bbox$是模板图像$\boldsymbol{z}$中目标的边界框，$stride$是采用的特征提取网络的总步长，一般为 8。在实际的实现过程中，RoIAlign 操作将会得到中间特征$\boldsymbol{r}' \in \mathbb{R}^{b\times c\times 4\times 4}$，之后再使用一层卷积层 Conv 得到最终的 RoIAlign 特征$\boldsymbol{r} \in \mathbb{R}^{b\times c\times 1\times 1}$。

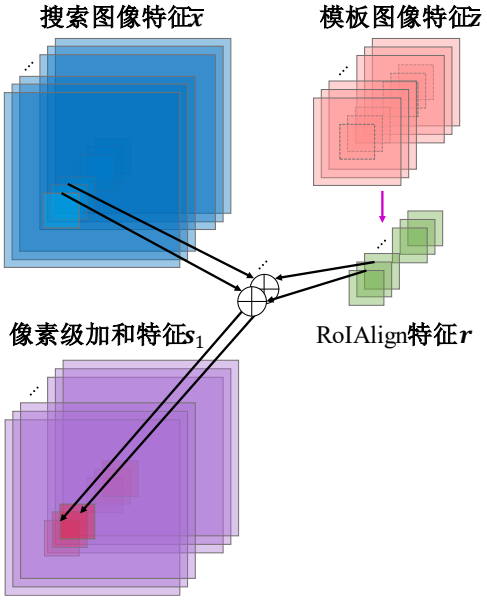

图 23 像素级加和操作

为了让搜索图像特征$\bar{\boldsymbol{x}}$的每一个 token 都与 RoIAlign 特征$\boldsymbol{r}$进行融合，这里采用广播求和机制，即将$\boldsymbol{r}$复制$H_x \times W_x$份与$\bar{\boldsymbol{x}}$相加：

$$\boldsymbol{s}_1 = (\bar{\boldsymbol{x}}) +_b (\boldsymbol{r}) \tag{3.4}$$

其中$+_b$是广播加和。

最终得到像素级加和特征$\boldsymbol{s}_1 \in \mathbb{R}^{b\times c\times H_x\times W_x}$。

3.2.2 相关性匹配操作

如图 24 所示，相关性匹配操作的作用是将模板图像特征$\bar{\boldsymbol{z}}$与像素级加和特征$\boldsymbol{s}_1$进行一次像素级的相关运算，即将模板图像特征中的每个 token 都与搜索图像特征中的所有 token 衡量相似性，以充分计算两种 token 之间的相关性。

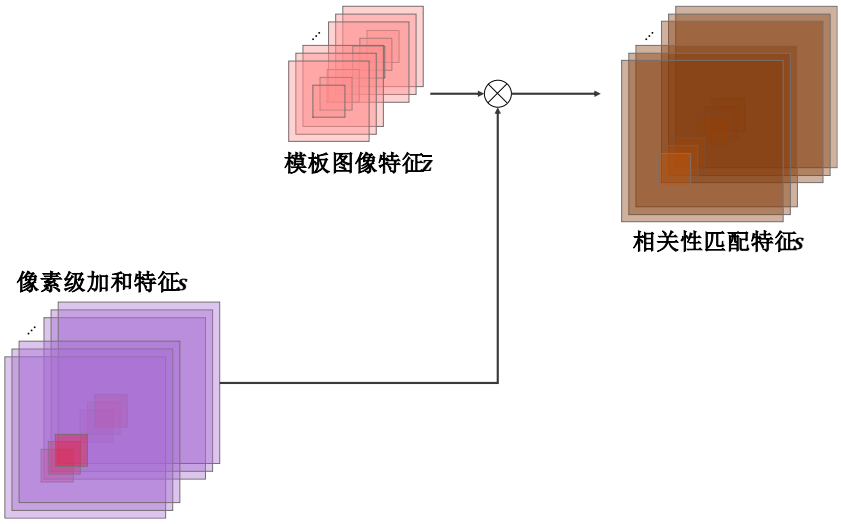

图 24　相关性匹配操作

不妨设有模板图像特征$\bar{\boldsymbol{z}} \in \mathbb{R}^{b\times c\times H_z\times W_z}$和像素级加和特征$\boldsymbol{s}_1 \in \mathbb{R}^{b\times c\times H_x\times W_x}$，则相关性匹配操作为：

$$\boldsymbol{s}_2 = \mathrm{Reshape}\Big(\mathrm{Reshape}(\bar{\boldsymbol{z}})^{\mathrm{T}}\mathrm{Reshape}(\boldsymbol{s}_1)\Big) \tag{3.5}$$

其中，Reshape 是指对四维特征张量的任意一种变形操作，$\mathrm{Reshape}(\bar{\boldsymbol{z}}) \in \mathbb{R}^{b\times c\times (H_z\times W_z)}$，$\mathrm{Reshape}(\boldsymbol{s}_1) \in \mathbb{R}^{b\times c\times (H_x\times W_x)}$，转置操作不影响 Batch 所在的维度，因此$\mathrm{Reshape}(\bar{\boldsymbol{z}})^{\mathrm{T}}\mathrm{Reshape}(\boldsymbol{s}_1) \in \mathbb{R}^{b\times (H_z\times W_z)\times (H_x\times W_x)}$，最后一次 Reshape 操作将$(H_x \times W_x)$拆分，最后得到的相关性匹配特征$\boldsymbol{s}_2 \in \mathbb{R}^{b\times (H_z\times W_z)\times H_x\times W_x}$。

3.2.3 仿射变换操作

如图 25 所示，仿射操作的思路源于视觉推理，通过对神经网络的“中间特征”应

用仿射变换来自适应地影响网络的输出。在本文中，仿射操作利用 RoIAlign 特征$\boldsymbol{r}$经过卷积产生仿射变换的权重$\boldsymbol{w}$和偏置$\boldsymbol{b}$，再利用权重和偏置对相关性匹配特征$\boldsymbol{s}_2$进行仿射变换。

不妨设有 RoIAlign 特征$\boldsymbol{r} \in \mathbb{R}^{b\times c\times 1\times 1}$和相关性匹配特征$\boldsymbol{s}_2 \in \mathbb{R}^{b\times (H_z\times W_z)\times H_x\times W_x}$，则仿射变换操作为：

$$\begin{cases} \boldsymbol{s}_2' = \mathrm{Conv}_1(\boldsymbol{s}_2) \in \mathbb{R}^{b\times c\times H_x\times W_x} \\ \boldsymbol{w} = \mathrm{Conv}_2(\boldsymbol{r}) \in \mathbb{R}^{b\times c\times 1\times 1} \\ \boldsymbol{b} = \mathrm{Conv}_3(\boldsymbol{r}) \in \mathbb{R}^{b\times c\times 1\times 1} \\ \boldsymbol{s}_3 = w \cdot_b \boldsymbol{s}_2' +_b \boldsymbol{b} \end{cases} \tag{3.6}$$

其中，$\mathrm{Conv}_{1,2,3}$都是待训练的二维卷积，$\cdot_b$为广播乘积运算，即1×1的$\boldsymbol{w}$与$H_x \times W_x$个$\boldsymbol{s}_2'$的 token 都进行乘积，$+_b$为广播求和，作用同理。

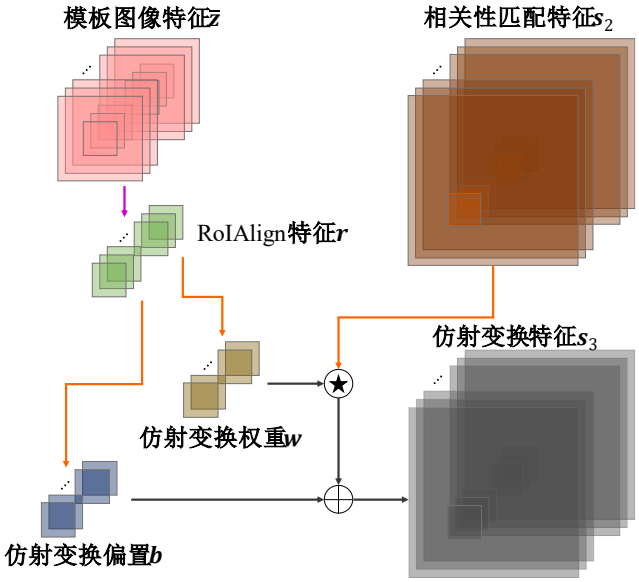

图 25 相关性匹配操作

仿射变换最终能够利用含有目标信息的$\boldsymbol{r}$对相关性匹配特征$\boldsymbol{s}_2$中的值进行动态的线性调整，使得相关性匹配的值在一个合理的数值范围内，最后得到仿射变换特征$\boldsymbol{s}_3 \in \mathbb{R}^{b\times c\times H_x\times W_x}$作为预相关运算模块的输出。

3.3 多层感知机相关运算模块

多层感知机相关运算模块的基本处理流程如图 26 所示，其中的所有卷积是1×1卷积，功能与全连接层或者线性层一致，可以使用全连接层替代实现。

在多层感知机相关运算模块中，特征张量的每一个 token 都被看作是一束“波

（wave）”，相关运算建立在这种波的混合之中。

由于波的特性，相位相近的波将会互相增强，相位相反的波将会互相削弱，因此相位对于波的自适应融合起到了类似于注意力机制的效果：不依靠查询算子、键算子、值算子，而是仅依靠相位的学习就可以自适应地调节各波之间地增强削弱关系。这相当于去掉了传统的注意力算子，可以大大减少计算量。

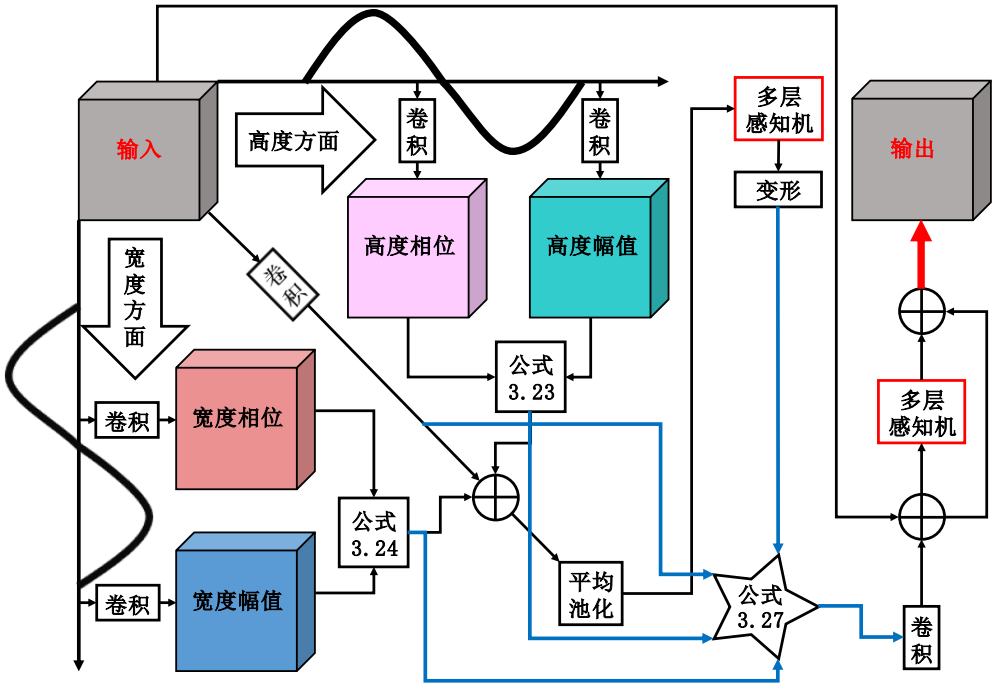

图 26　多层感知机相关运算模块

3.3.1 波的概述

文献[21]首次将图像或者图像特征的每一个 token 看作是一种波，并在此基础上设计了特征提取网络，在 ImageNet 图像分类任务上取得了很好的表现，从实验的角度证明了这种思路的可行性。

如图 27 所示，记一图像的特征张量$\boldsymbol{z} \in \mathbb{R}^{c\times H\times W}$，则特征张量$\boldsymbol{z}$在空间维度可以拆分为$n$个 token：

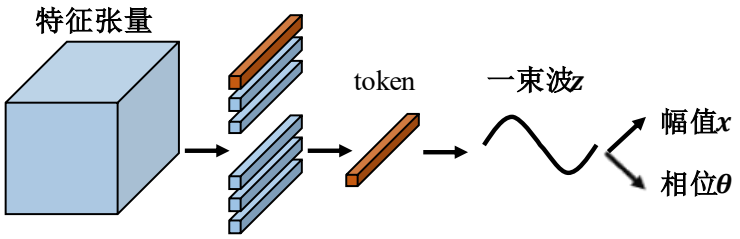

图 27　token 是一种波

$$\boldsymbol{z}=\{\boldsymbol{z}_1,\boldsymbol{z}_2,\cdots,\boldsymbol{z}_n\},\ \ \boldsymbol{z}_j\in\mathbb{R}^c,\ \ j=1,2,\cdots,n$$

其中$n=H\times W$。

若将每一个 token 看作是一束波，则有：

$$\boldsymbol{z}_j=\boldsymbol{x}_j\odot e^{i\boldsymbol{\theta}_j},\ \ j=1,2,\cdots,n \tag{3.7}$$

其中i是虚数单位，满足$i^2=-1$，e是是自然对数的底数，$\odot$是元素乘。

$\boldsymbol{x}_j=|\boldsymbol{z}_j|$是该波的幅值，是一个实非负值或者实非负张量，其中蕴含该 token 的内容（context）信息；$\boldsymbol{\theta}_j$是该波的相位，代表该 token 的位置（Location）信息。

考虑两束波$\boldsymbol{z}_1$和$\boldsymbol{z}_2$的叠加，其实质上就是两个 token 的融合，则在数学推导上有$\boldsymbol{z}_+=\boldsymbol{z}_1+\boldsymbol{z}_2$仍旧是一束波，它的幅值$|\boldsymbol{z}_+|$和相位$\boldsymbol{\theta}_+$计算公式如 3.8 和 3.9 所示：

$$|\boldsymbol{z}_+|=|\boldsymbol{z}_1+\boldsymbol{z}_2|=\sqrt{|\boldsymbol{z}_1|^2+|\boldsymbol{z}_2|^2+2|\boldsymbol{z}_1|\odot|\boldsymbol{z}_2|\odot\cos(\boldsymbol{\theta}_1-\boldsymbol{\theta}_2)} \tag{3.8}$$

$$\boldsymbol{\theta}_+=\boldsymbol{\theta}_1+\mathrm{atan2}(|\boldsymbol{z}_2|\odot\sin(\boldsymbol{\theta}_2-\boldsymbol{\theta}_1),|\boldsymbol{z}_1|+|\boldsymbol{z}_2|\odot\cos(\boldsymbol{\theta}_2-\boldsymbol{\theta}_1)) \tag{3.9}$$

其中$\mathrm{atan2}(\cdot,\cdot)$是两个变量的反正切函数。

由上述公式可以看出，$\boldsymbol{\theta}_1$和$\boldsymbol{\theta}_2$的关系将会影响$|\boldsymbol{z}_+|$的大小。对于周期性的余弦函数而言：

当$\boldsymbol{\theta}_2=\boldsymbol{\theta}_1+2\pi*m$时，$\cos(\boldsymbol{\theta}_1-\boldsymbol{\theta}_2)$取得最大值 1，$\boldsymbol{z}_1$和$\boldsymbol{z}_2$的叠加将会相互增强，叠加产生的新波幅值较大，其中$m\in[0,\pm2,\pm4,\cdots]$。

当$\boldsymbol{\theta}_2=\boldsymbol{\theta}_1+\pi*m$时，$\cos(\boldsymbol{\theta}_1-\boldsymbol{\theta}_2)$取得最小值-1，$\boldsymbol{z}_1$和$\boldsymbol{z}_2$的叠加将会相互削弱，叠加产生的新波幅值较小，其中$m\in[\pm1,\pm3,\cdots]$。

如果能够自适应地学习$\boldsymbol{\theta}$的表征，则可以自适应地影响 tokens 之间的融合效果。对于任一束波$\boldsymbol{z}_j=|\boldsymbol{z}_j|\odot e^{i\boldsymbol{\theta}_j}$而言，相位$\boldsymbol{\theta}_j$的学习还可以去除幅值$\boldsymbol{x}_j=|\boldsymbol{z}_j|$的非负性约束，这是因为非负性约束可以被包含在$\boldsymbol{\theta}$的学习之中：

$$|\boldsymbol{z}_j|\odot e^{i\boldsymbol{\theta}_j}=\begin{cases}\boldsymbol{z}_j\odot e^{i\boldsymbol{\theta}_j}, & \boldsymbol{z}_j>0\\ \boldsymbol{z}_j\odot e^{i(\boldsymbol{\theta}_j+\pi)}, & \boldsymbol{z}_j\le0\end{cases} \tag{3.10}$$

3.3.2 相位感知注意力机制

图像特征$\boldsymbol{z}=\{\boldsymbol{z}_1,\boldsymbol{z}_2,\cdots,\boldsymbol{z}_n\}$的每一个 token 看作是一束波，本文设计该波的幅值和相位可以从数据中学习得到，即：

$$\begin{cases} \boldsymbol{x}_j = \mathrm{Conv}_x(\boldsymbol{z}_j), & j = 1,2,\cdots,n \\ \boldsymbol{\theta}_j = \mathrm{Conv}_\theta(\boldsymbol{z}_j), & j = 1,2,\cdots,n \end{cases} \tag{3.11}$$

其中$\boldsymbol{z}_j \in \mathbb{R}^{c\times 1\times 1}$，因此$Conv_{x,\theta}$均是作用在通道维度的$1\times 1$卷积，可以用全连接层或者线性层替代实现：

$$\begin{cases} \boldsymbol{x}_j = \mathrm{Channel\_FC}(\boldsymbol{z}_j, \boldsymbol{W}_x^c), & j = 1,2,\cdots,n \\ \boldsymbol{\theta}_j = \mathrm{Channel\_FC}(\boldsymbol{z}_j, \boldsymbol{W}_\theta^c), & j = 1,2,\cdots,n \end{cases} \tag{3.12}$$

其中$\boldsymbol{W}_{x,\theta}^c$是参数可学习的权重矩阵，右上角标$c$代表这是Channel_FC的参数。

如前所述，注意力机制的使用在孪生网络目标跟踪的相关运算技术中是有用的，而将图像 token 或者图像特征 token 看作是波，并对波的相位进行学习，便可以利用各波之间的相位关系，自适应地产生不同的波的叠加效果，本质上即实现了 token 之间自适应的融合操作。

为了使用全连接层实现 token 之间的融合，即波的叠加，需要对波的表示利用欧拉公式展开：

$$\boldsymbol{z}_j = \boldsymbol{x}_j \odot e^{i\boldsymbol{\theta}_j} = \boldsymbol{x}_j \odot \cos\boldsymbol{\theta}_j + i\boldsymbol{x}_j \odot \sin\boldsymbol{\theta}_j, \quad j = 1,2,\cdots,n \tag{3.13}$$

其中i是虚数单位，$\odot$是元素乘。

则 token 之间的融合可以用下式表示：

$$\boldsymbol{o}_j = \sum\nolimits_k \left(W_{jk}^{cos}\boldsymbol{x}_k \odot \cos\boldsymbol{\theta}_k + iW_{jk}^{sin}\boldsymbol{x}_k \odot \sin\boldsymbol{\theta}_k\right), \quad j = 1,2,\cdots,n \tag{3.14}$$

然而由于虚数单位i的存在，$\boldsymbol{o}_j$是一个复数，难以用一般的神经网络进行学习。文献**错误!未找到引用源。**指出，采用一些量子测量方法如通常测量，可以将复值表征投影到实值观测，参考这种思路可以将上式投影至实值再进行 token 之间的融合。

对于一束已经经过欧拉公式展开，分别使用正弦函数、余弦函数进行表示的波，它的实值投影可以直接写出：

$$\boldsymbol{o}_j = \mathrm{Token\_FC}(\boldsymbol{z}, \boldsymbol{W}^t) = \sum\nolimits_k \left(W_{jk}^{cos}\boldsymbol{x}_k \odot \cos\boldsymbol{\theta}_k + W_{jk}^{sin}\boldsymbol{x}_k \odot \sin\boldsymbol{\theta}_k\right) \tag{3.15}$$

其中W_{jk}^{cos}和W_{jk}^{sin}是两个参数可学习的权重矩阵，$\odot$是元素乘。

相位感知注意力机制的学习总结为输入$\boldsymbol{z}_j$，输出为$\boldsymbol{o}_j$，从输入到输出只需经历如式 3.16 所示的三种线性操作，其中不含任何显式的注意力算子的计算，从原理上减少了计算量：

$$\begin{cases} \boldsymbol{x}_j = \text{Channel\_FC}\left(\boldsymbol{z}_j, \boldsymbol{W}_x^c\right) \\ \boldsymbol{\theta}_j = \text{Channel\_FC}\left(\boldsymbol{z}_j, \boldsymbol{W}_\theta^c\right) \\ \boldsymbol{o}_j = \sum_k \left(W_{jk}^{cos} \boldsymbol{x}_k \odot \cos \boldsymbol{\theta}_k + W_{jk}^{sin} \boldsymbol{x}_k \odot \sin \boldsymbol{\theta}_k\right) \end{cases} \tag{3.16}$$

其中需要经过训练从数据中学习得到的权重为$\boldsymbol{W}_x^c, \boldsymbol{W}_\theta^c, \boldsymbol{W}^{cos}, \boldsymbol{W}^{sin}$。

3.3.3 相关运算模块

基于相位感知注意力机制的多层感知机相关运算模块的功能是将预相关运算模块的初步相关运算结果再进行空间维度的深度相关运算。它接收预相关运算模块的输出，即仿射变换特征$\boldsymbol{s}_3 \in \mathbb{R}^{b\times c\times H_x\times W_x}$，输出深度相关运算结果$\boldsymbol{o} \in \mathbb{R}^{b\times c\times H_x\times W_x}$。

由于该模块中相位感知注意力机制处理的对象不是一维特征，而是二维图像特征，因此将特征张量$\boldsymbol{s}_3$作为$wave_{in}$并分别按高（High）、宽（Width）和通道（Channel）进行处理。最后深度相关运算结果$\boldsymbol{o}$作为基于相位感知注意力机制的多层感知机相关运算模块的输出$wave_{out}$。

如图 26 所示，本章设计的基于相位感知注意力机制的多层感知机相关运算模块分为下述 6 个步骤进行：

(1) 幅值和相位的表征：

分别按高和宽对$wave_{in} \in \mathbb{R}^{b\times c\times H_x\times W_x}$采用全连接层进行表征：

按高度表征幅值和按高度表征相位：

$$\boldsymbol{x}_j^h = \text{Channel\_FC}\left(\boldsymbol{z}_j, \boldsymbol{W}_x^h\right), \quad j = 1,2,\cdots,n \tag{3.17}$$

$$\boldsymbol{\theta}_j^h = \text{Channel\_FC}\left(\boldsymbol{z}_j, \boldsymbol{W}_\theta^h\right), \quad j = 1,2,\cdots,n \tag{3.18}$$

按宽度表征幅值和按宽度表征相位：

$$\boldsymbol{x}_j^w = \text{Channel\_FC}\left(\boldsymbol{z}_j, \boldsymbol{W}_x^w\right), \quad j = 1,2,\cdots,n \tag{3.19}$$

$$\boldsymbol{\theta}_j^w = \text{Channel\_FC}\left(\boldsymbol{z}_j, \boldsymbol{W}_\theta^w\right), \quad j = 1,2,\cdots,n \tag{3.20}$$

(2) 相位感知注意力机制融合：

公式 3.15 可以转换成分别对高度和宽度的各波进行叠加

按高度采用相位感知注意力叠加可以得到高度融合特征$\boldsymbol{h}$：

$$\boldsymbol{h}_j = \sum_k \begin{bmatrix} W_{jk}^{h,cos} & W_{jk}^{h,sin} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k^h \odot \cos \boldsymbol{\theta}_k^h \\ \boldsymbol{x}_k^h \odot \sin \boldsymbol{\theta}_k^h \end{bmatrix}, \quad j = 1,2,\cdots,n \tag{3.21}$$

按宽度采用相位感知注意力叠加可以得到宽度融合特征$\boldsymbol{w}$：

$$\boldsymbol{w}_j = \sum_k \begin{bmatrix} W_{jk}^{w,cos} & W_{jk}^{w,sin} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k^w \odot \cos \boldsymbol{\theta}_k^w \\ \boldsymbol{x}_k^w \odot \sin \boldsymbol{\theta}_k^w \end{bmatrix} \quad j \qquad \cdots, n \tag{3.22}$$

可以将$\boldsymbol{h} \in \mathbb{R}^{b \times c \times H_x \times W_x}$的学习写成大矩阵的形式：

$$\boldsymbol{h} = \begin{bmatrix} \boldsymbol{h}_1 \\ \boldsymbol{h}_2 \\ \vdots \\ \boldsymbol{h}_n \end{bmatrix} \begin{matrix} W_{11}^{h,cos} & W_{12}^{h,cos} & \cdots & W_{1n}^{h,cos} & W_{11}^{h,sin} & W_{12}^{h,sin} & \cdots & W_{1n}^{h,sin} \\ W_{21}^{h,cos} & W_{22}^{h,cos} & \cdots & W_{2n}^{h,cos} & W_{21}^{h,sin} & W_{22}^{h,sin} & \cdots & W_{2n}^{h,sin} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ W_{n1}^{h,cos} & W_{n2}^{h,cos} & \cdots & W_{nn}^{h,cos} & W_{n1}^{h,sin} & W_{n2}^{h,sin} & \cdots & W_{nn}^{h,sin} \end{matrix} \begin{matrix} \boldsymbol{x}_1^h \odot \cos \boldsymbol{\theta}_1^h \\ \boldsymbol{x}_2^h \odot \cos \boldsymbol{\theta}_2^h \\ \vdots \\ \boldsymbol{x}_n^h \odot \cos \boldsymbol{\theta}_n^h \\ \boldsymbol{x}_1^h \odot \sin \boldsymbol{\theta}_1^h \\ \boldsymbol{x}_2^h \odot \sin \boldsymbol{\theta}_2^h \\ \vdots \\ \boldsymbol{x}_n^h \odot \sin \boldsymbol{\theta}_n^h \end{matrix}$$

$$\begin{matrix} W_{11}^{h,cos} & W_{12}^{h,cos} & \cdots & W_{1n}^{h,cos} & W_{11}^{h,sin} & W_{12}^{h,sin} & \cdots & W_{1n}^{h,sin} \\ W_{21}^{h,cos} & W_{22}^{h,cos} & \cdots & W_{2n}^{h,cos} & W_{21}^{h,sin} & W_{22}^{h,sin} & \cdots & W_{2n}^{h,sin} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ W_{n1}^{h,cos} & W_{n2}^{h,cos} & \cdots & W_{nn}^{h,cos} & W_{n1}^{h,sin} & W_{n2}^{h,sin} & \cdots & W_{nn}^{h,sin} \end{matrix} \left(\begin{matrix} \boldsymbol{x}_1^h \\ \boldsymbol{x}_2^h \\ \vdots \\ \boldsymbol{x}_n^h \\ \boldsymbol{x}_1^h \\ \boldsymbol{x}_2^h \\ \vdots \\ \boldsymbol{x}_n^h \end{matrix} \odot \begin{matrix} \cos \boldsymbol{\theta}_1^h \\ \cos \boldsymbol{\theta}_2^h \\ \vdots \\ \cos \boldsymbol{\theta}_n^h \\ \sin \boldsymbol{\theta}_1^h \\ \sin \boldsymbol{\theta}_2^h \\ \vdots \\ \sin \boldsymbol{\theta}_n^h \end{matrix} \right)$$

$$\begin{bmatrix} \boldsymbol{W}^{h,cos} & \boldsymbol{W}^{h,sin} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}^h \odot \cos \boldsymbol{\theta}^h \\ \boldsymbol{x}^h \odot \sin \boldsymbol{\theta}^h \end{bmatrix} \tag{3.23}$$

其中$n = H_x \times W_x$，$\boldsymbol{h}_j \in \mathbb{R}^{b \times c}$，$\odot$是元素乘，$\boldsymbol{W}^{h,cos}$和$\boldsymbol{W}^{h,sin}$是待训练的参数矩阵。

从上述高度融合特征$\boldsymbol{h}$的大矩阵形式可以看出，将各 token 采用相位感知注意力进行叠加，可以使用全连接层进行实现，具体的实现可以采用$\boldsymbol{x}^h \odot \cos \boldsymbol{\theta}^h$与$\boldsymbol{x}^h \odot \sin \boldsymbol{\theta}^h$先进行拼接操作，再使用$1 \times 1$卷积融合特征。

同理可以得到宽度融合特征$\boldsymbol{w}$的大矩阵形式：

$$\boldsymbol{w} = \begin{bmatrix} \boldsymbol{w}_1 \\ \boldsymbol{w}_2 \\ \vdots \\ \boldsymbol{w}_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{W}^{w,cos} & \boldsymbol{W}^{w,sin} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}^w \odot \cos \boldsymbol{\theta}^w \\ \boldsymbol{x}^w \odot \sin \boldsymbol{\theta}^w \end{bmatrix} \in \mathbb{R}^{b \times c \times H_x \times W_x} \tag{3.24}$$

(3) 通道融合特征的表征：

采用全连接层对输入特征$wave_{in}$的通道特征进行融合：

$$\boldsymbol{c}_j = \text{Channel\_FC}\left(\boldsymbol{z}_j, \boldsymbol{W}^c\right), \quad j \qquad , \cdots, n \tag{3.25}$$

得到通道融合特征$\boldsymbol{c} \in \mathbb{R}^{b \times c \times H_x \times W_x}$。

(4) 系数特征$\boldsymbol{a}$的计算：

为了进一步动态调整高度融合特征$\boldsymbol{h}$、宽度融合特征$\boldsymbol{w}$和通道融合特征$\boldsymbol{c}$的值，需要

计算系数特征$\boldsymbol{a}$，具体计算过程如下：

$$\boldsymbol{a} = \text{Reshape}\left(\text{MLP}_1\left(\text{adaptive\_avg\_pool2d}(\boldsymbol{h}+\boldsymbol{w}+\boldsymbol{c})\right)\right) \tag{3.26}$$

其中，adaptive_avg_pool2d是二维自适应平均池化运算，可以将$H_x \times W_x$的尺寸池化到$\times 1$的尺寸，即$\text{adaptive_avg_pool2d}(\boldsymbol{h}+\boldsymbol{w}+\boldsymbol{c}) \in \mathbb{R}^{b\times c\times 1\times 1}$。

这里使用的 MLP 如图 28 所示，它采用类似于文献[40]中瓶颈（Bottleneck）结构的设计，即先使用1×1卷积进行特征降维，再使用1×1卷积进行特征升维。

图 28 中$\text{Conv2d}(c,\ c//4,\ 1,\ 1,\ 0)$是指该二维卷积层的输入维度为$c$，输出维度是$c$除以 4 并向下取整，卷积核尺寸为$1\times 1$，步长为$1\times 1$，不进行填充（padding）操作；另一个卷积层同理。

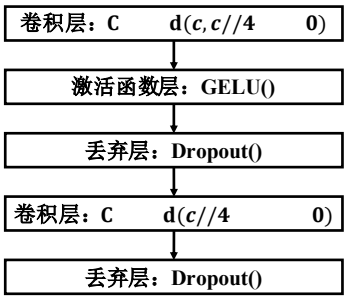

图 28 MLP 相关运算模块中的第一个 MLP

经过该 MLP 后输出的特征张量属于$\mathbb{R}^{b\times 3c\times 1\times 1}$，再经过变形操作可以得到系数特征$\boldsymbol{a} \in \mathbb{R}^{3\times b\times c\times 1\times 1}$，这是一个五维张量。

(5) 系数校准特征$\boldsymbol{o}'$的计算：

使用系数特征$\boldsymbol{a}$对高度融合特征$\boldsymbol{h}$、宽度融合特征$\boldsymbol{w}$和通道融合特征$\boldsymbol{c}$进行加权求和，并出于增强表达能力的考虑，再添加一次通道维度的全连接操作：

$$\boldsymbol{o}'_j = \text{Channel\_FC}(\boldsymbol{h}\odot\boldsymbol{a}[0] + \boldsymbol{w}\odot\boldsymbol{a}[1] + \boldsymbol{c}\odot\boldsymbol{a}[2], \boldsymbol{W}_o^c) \tag{3.27}$$

其中，$\boldsymbol{a}[0]$是系数特征$\boldsymbol{a}$的第一个维度，$\boldsymbol{a}[1]$是系数特征$\boldsymbol{a}$的第二个维度，$\boldsymbol{a}[2]$是系数特征$\boldsymbol{a}$的第三个维度，$\boldsymbol{a}[0], \boldsymbol{a}[1], \boldsymbol{a}[2] \in \mathbb{R}^{b\times c\times 1\times 1}$，$\odot$是元素乘，$\boldsymbol{W}_o^c$是带学习的参数矩阵，系数校准特征$\boldsymbol{o}' \in \mathbb{R}^{b\times c\times H_x\times W_x}$。

(6) 深度相关运算结果$\boldsymbol{o}$的计算：

文献[40]指出残差连接结构的有效性，因此深度相关运算结果$\boldsymbol{o}$的计算也进行两次残差连接。

$$\begin{cases} \boldsymbol{o}'' = wave_{in} + \boldsymbol{o}' \in \mathbb{R}^{b\times c\times H_x\times W_x} \\ wave_{out} = \boldsymbol{o} = \boldsymbol{o}'' + \text{MLP}_2(\boldsymbol{o}'') \end{cases} \tag{3.28}$$

其中$\boldsymbol{o}''$再经过一个 MLP。这里使用的 MLP 如图 29 所示，它采用类似于文献[41]中倒置残差（Inverted Residuals）结构的设计，即先使用1×1卷积进行特征升维，再使用1×1卷积进行特征降维。

图 29 中$\mathrm{Conv2d}(c,\ 4c,\ 1,\ 1,\ 0)$是指该二维卷积层的输入维度为$c$，输出维度是 4 倍的$c$，卷积核尺寸为$1\times1$，步长为$1\times1$，不进行填充（padding）操作；另一个卷积层同理可得其解释。

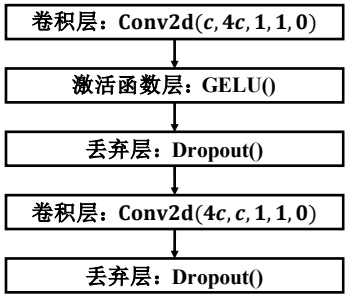

图 29　MLP 相关运算模块中的第二个 MLP

深度相关运算结果$\boldsymbol{o}\in\mathbb{R}^{b\times c\times H_x\times W_x}$作为基于相位感知注意力机制的多层感知机相关运算模块的输出$wave_{out}$。

3.4 实验与分析

为了验证本章的孪生网络目标跟踪算法（称为“SiamWAVE”）以及设计的相关运算模块的有效性，实验分为表 2 中七种模型配置：

表 2　七种模型配置

模型配置	layer0	layer1	layer2	layer3	layer4	layer5	layer6

其中，layer0 是指仅使用预相关运算模块进行相关运算，layer1 到 layer6 是指在预相关运算模块的基础上堆叠对应数量的多层感知机相关运算模块。

七种配置的模型在全部 GOT-10k、YT-BB、COCO、DET 和 VID 五个训练集上进行模型训练，一共训练 32 轮（epoch），在前 10 轮冻结骨干网络使其参数在训练过程中不变，从第 11 轮开始对骨干网络和相关运算模块一并训练。七种配置的模型在 OTB100、GOT-10k 和 LaSOT 三个测试集上进行性能评估。

3.4.1 OTB100 测试集实验结果

(1) 算法评比

本章设计的基于相位感知注意力空间特征融合的孪生网络跟踪算法在 OTB100 数据集上与近年各大顶级会议上发表的跟踪算法进行指标评比结果如表 3 与图 30 所示，其中表 3 和图 30 均展示成功率和精确率两项指标，各指标的解释详见 2.3 节。

表 3 SiamWAVE 与各跟踪算法在 OTB100 测试集的指标评比结果

跟踪算法名称	OTB100 测试集	
	成功率（%）	精确率（%）
SAOT[42]	**71.4**	**92.6**
AutoMatch[38]	**71.4**	**92.6**
SiamGAT[16]	70.9	91.6
SiamRPN++[12]	69.6	92.3
AiATrack[43]	69.6	91.7
PrDimp50[44]	69.5	89.7
TransT[20]	69.1	89.3
PGNet[15]	69.1	89.2
SiamWAVE(layer6)	**68.8**	**88.8**
SPM[45]	68.7	89.9
Ocean[46]	68.4	92.0
SiamWAVE(layer4)	**68.6**	**88.8**
DiMP[47]	68.4	-
SiamFC++[48]	68.3	-
ROAM[49]	68.1	90.8
SiamWAVE(layer3)	**67.7**	**87.1**
SiamWAVE(layer5)	**67.5**	**87.1**
SiamDW[11]	**67.4**	-
SiamWAVE(layer2)	**67.1**	**87.8**
ATOM[50]	66.7	87.9
SiamWAVE(layer1)	**66.4**	**86.3**
SiamRPN[51]	63.7	85.1

从图表中的结果可以看出，本章算法在 OTB100 数据集上的表现具有竞争力：该算法在 layer6 配置情况下相较于基线算法 SiamDW[11]的成功率指标（表中蓝色标出）有了 1%的效果提升，相较于其它算法也均在成功率与精确率上有了部分提升，这说明设计的相关运算模块在空间维度使用简单的多层感知机结构将模板图像特征和搜索图像特征进行融合也可以获得不输于卷积的性能表现。然而，由于本章设计的算法只考虑空间维

度的相关运算而忽略了通道维度的相关运算，因此它的表现略逊于既考虑通道维度相关运算又考虑空间维度相关运算的算法 PGNet[15]；由于本章设计的算法仅使用简单的多层感知机，因此它的表现逊于使用复杂沉重的 Transformer 结构设计相关运算模块的算法 TransT[20]、AiATrack[43]等，表 3 中表现最好的 SAOT[42]和 AutoMatch[38]算法及其指标用红色标出。

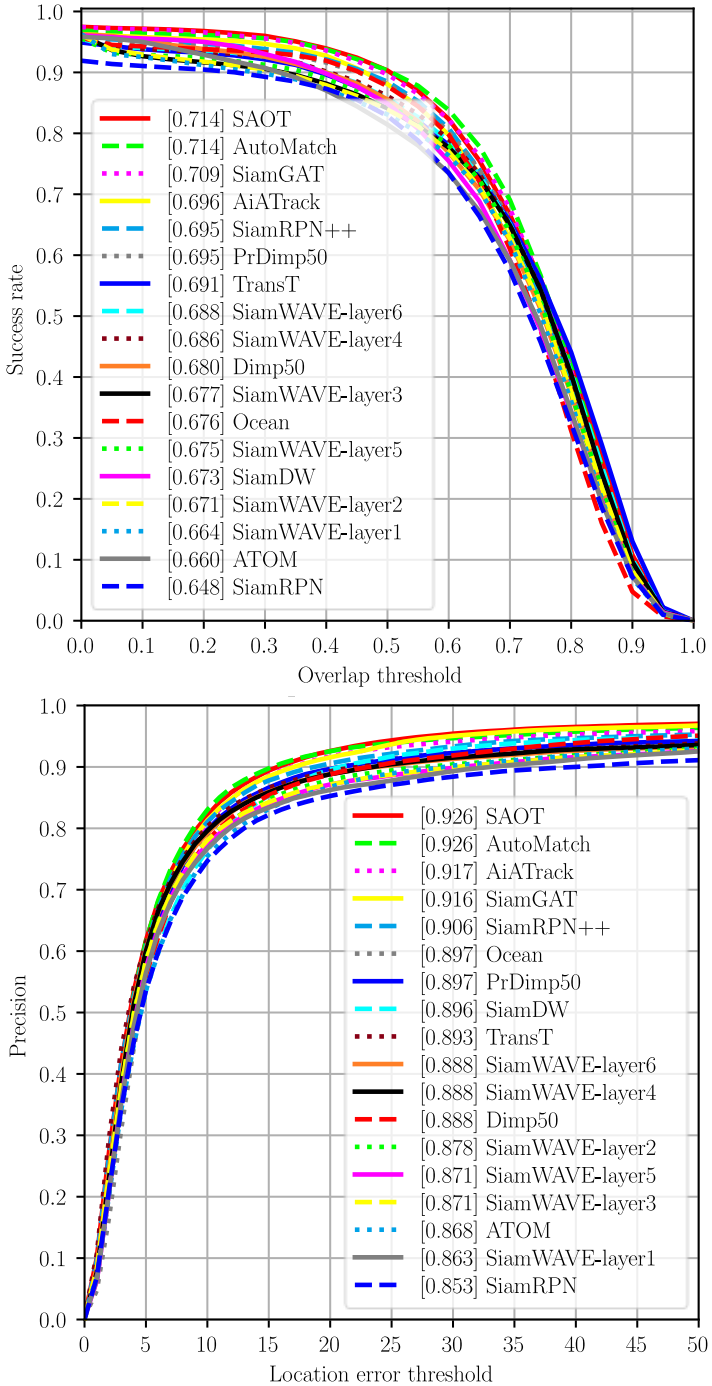

图 30　SiamWAVE 在 OTB100 测试集的成功率图（上）与精确图（下）

(2) 对比实验

表 3 也展示了本章算法的六种配置的表现，随着设计的相关运算模块的逐层添加，其在 OTB100 数据集上的性能表现也基本逐渐提升，这一定程度上表明了该相关运算模块的有效性。

如图 31 所示，为了进一步证明设计的相关运算模块的有效性，本章设计了对比实验：包括只加预相关运算模块（即 layer0）在内的全部七种配置，在第 11 到第 32 轮训练过程中的各轮成功率指标与精确率指标进行对比。

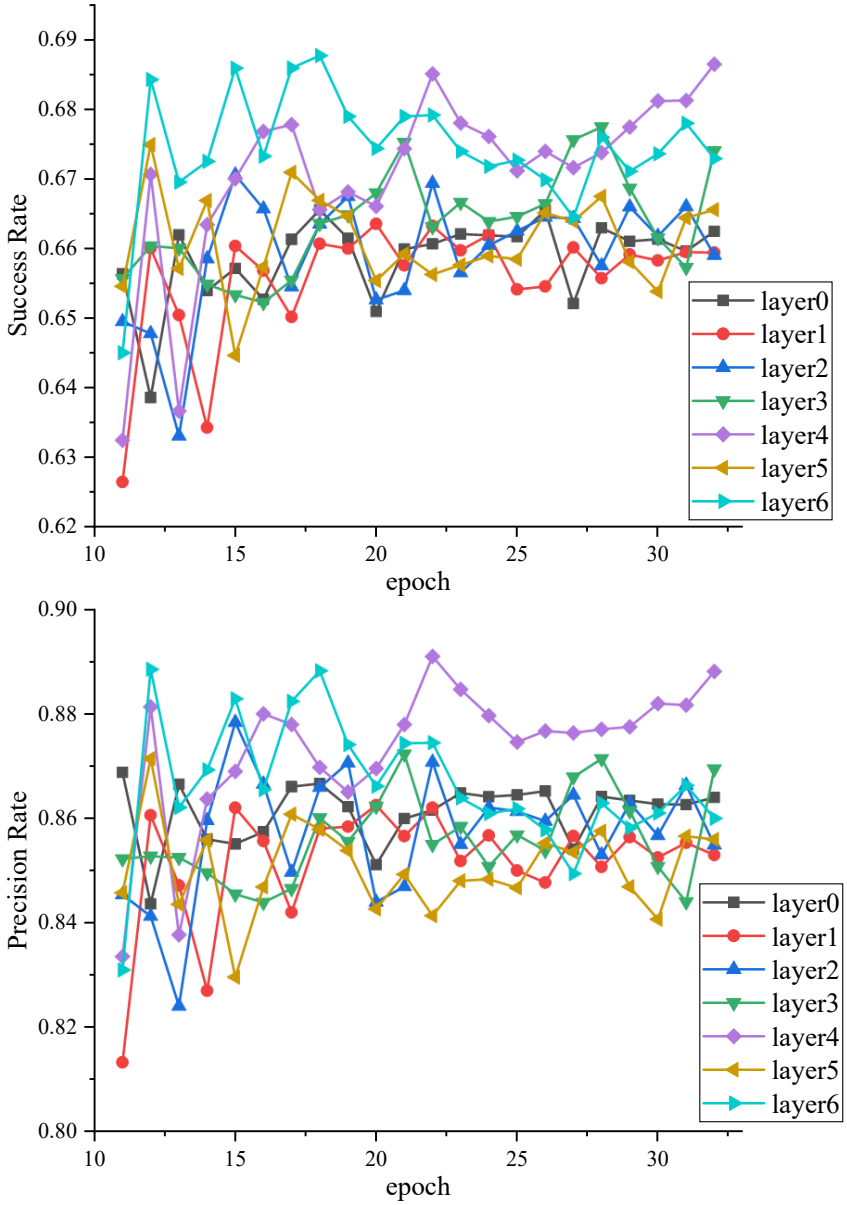

图 31 SiamWAVE 的七种配置在 OTB100 测试集的逐轮指标

该对比实验表明，简单地堆叠一至二层基于多层感知机的相关运算模块并不能取得相对于预相关运算模块的显著的性能提升；当层数继续增多时，基于多层感知机的相关运算模块逐渐表现出显著地性能提升效果。

3.4.2 GOT-10k 测试集实验结果

(1) 算法评比

本章设计的基于相位感知注意力空间特征融合的孪生网络跟踪算法在 GOT-10k 数据集上与近年各大顶级会议上发表的跟踪算法进行指标评比结果如表 4 与图 32 所示，其中表 4 列出 AO、$SR_{0.5}$ 和 $SR_{0.75}$ 三项指标，图 32 单独绘制 GOT-10k 数据集上的成功率图，各指标的解释详见 2.3 节。

表 4 SiamWAVE 与各跟踪算法在 GOT-10k 测试集的指标评比结果

跟踪算法名称	GOT-10k 测试集		
	AO（%）	$SR_{0.5}$（%）	$SR_{0.75}$（%）
AiATrack[43]	**69.6**	**80.0**	**63.2**
SPM[45]	68.7	89.9	
TransT[20]	67.1	76.8	60.9
SiamWAVE(layer3)	**66.1**	**76.4**	**57.3**
SiamWAVE(layer6)	**65.7**	**76.3**	**56.0**
SiamWAVE(layer5)	**65.4**	**75.4**	**55.8**
AutoMatch[38]	65.2	76.6	54.3
SiamWAVE(layer4)	**64.3**	**75.5**	**53.6**
SAOT[42]	64.0	74.9	-
PrDimp50[44]	63.4	73.8	54.3
SiamWAVE(layer2)	**63.2**	**73.5**	**52.2**
SiamGAT[16]	62.7	74.3	48.8
SiamWAVE(layer1)	**62.1**	**72.7**	**51.0**
Ocean[46]	61.1	72.1	-
DiMP[47]	61.1	71.7	49.2
SiamFC++[48]	59.5	69.5	47.9
ROAM[49]	46.5	53.2	23.6
SiamDW[11]	**41.6**	-	-
PGNet[15]	-	-	-
SiamRPN++[12]	-	-	-
ATOM[50]	-	-	-
SiamRPN[51]	-	-	-

从图表中的结果可以看出，本章算法在 GOT-10k 数据集上的表现具有很强的竞争力：该算法在 layer3 配置情况下相较于基线算法 SiamDW[11]的 AO 指标（表中蓝色标出）

有了近 25%的效果提升，相较于其它算法也均在 AO、$SR_{0.5}$ 和 $SR_{0.75}$ 三项指标上有了部分提升，这也说明设计的相关运算模块在空间维度使用简单的多层感知机结构将模板图像特征和搜索图像特征进行融合在一些数据集上可以获得远超卷积的性能表现。在该测试集上，本章设计的算法在只考虑空间维度的相关运算而忽略了通道维度的相关运算的情况下，展现出极佳的跟踪性能。但是由于本章设计的算法仅使用简单的多层感知机，因此它的表现逊于使用复杂沉重的 Transformer 结构设计相关运算模块的算法 TransT[20]、AiATrack[43]等，表 4 中表现最好的 AiATrack 算法的指标用红色标出。

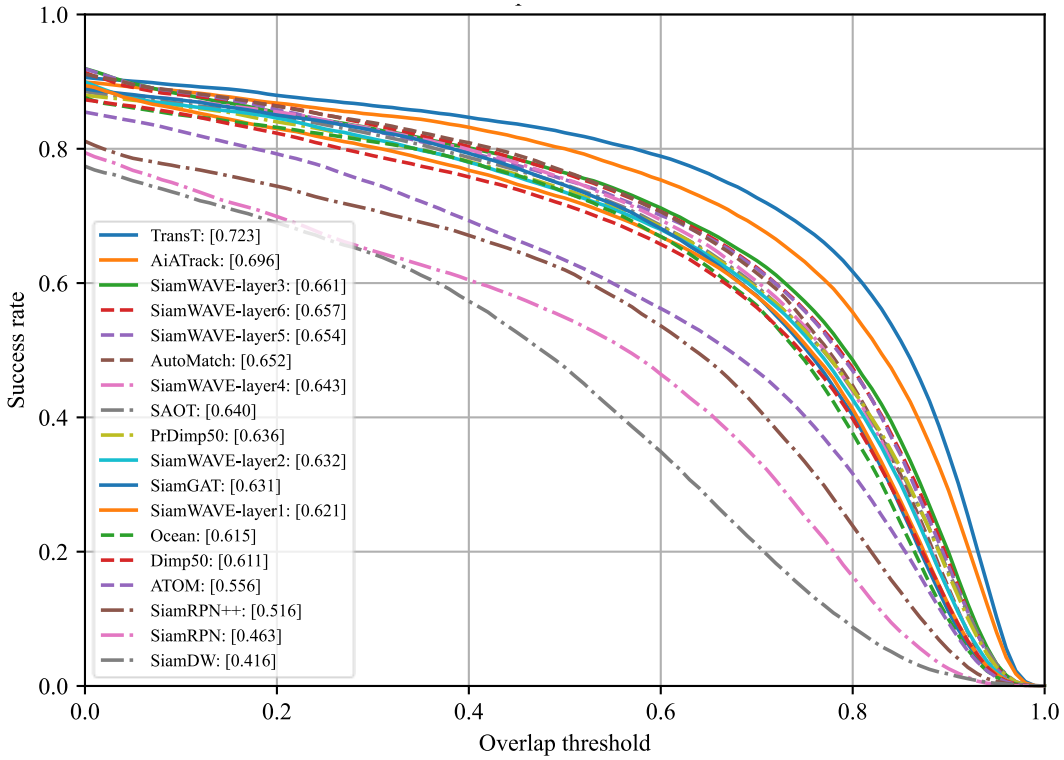

图 32 SiamWAVE 在 GOT-10k 测试集的成功率图

(2) 对比实验

表 4 也展示了本章算法的六种配置的表现，随着设计的相关运算模块的逐层添加，其在 GOT-10k 数据集上的性能表现也基本逐渐提升，这一定程度上表明了该相关运算模块的有效性。

如图 33 所示，为了进一步证明设计的相关运算模块的有效性，本章设计了对比实验：包括只加预相关运算模块（即 layer0）在内的全部七种配置，在第 11 到第 32 轮训练过程中的各轮 AO、$SR_{0.5}$ 和 $SR_{0.75}$ 指标进行对比。图中不同颜色的折线代表了不同的配置，横坐标为训练轮数（epoch），纵坐标为指标值。

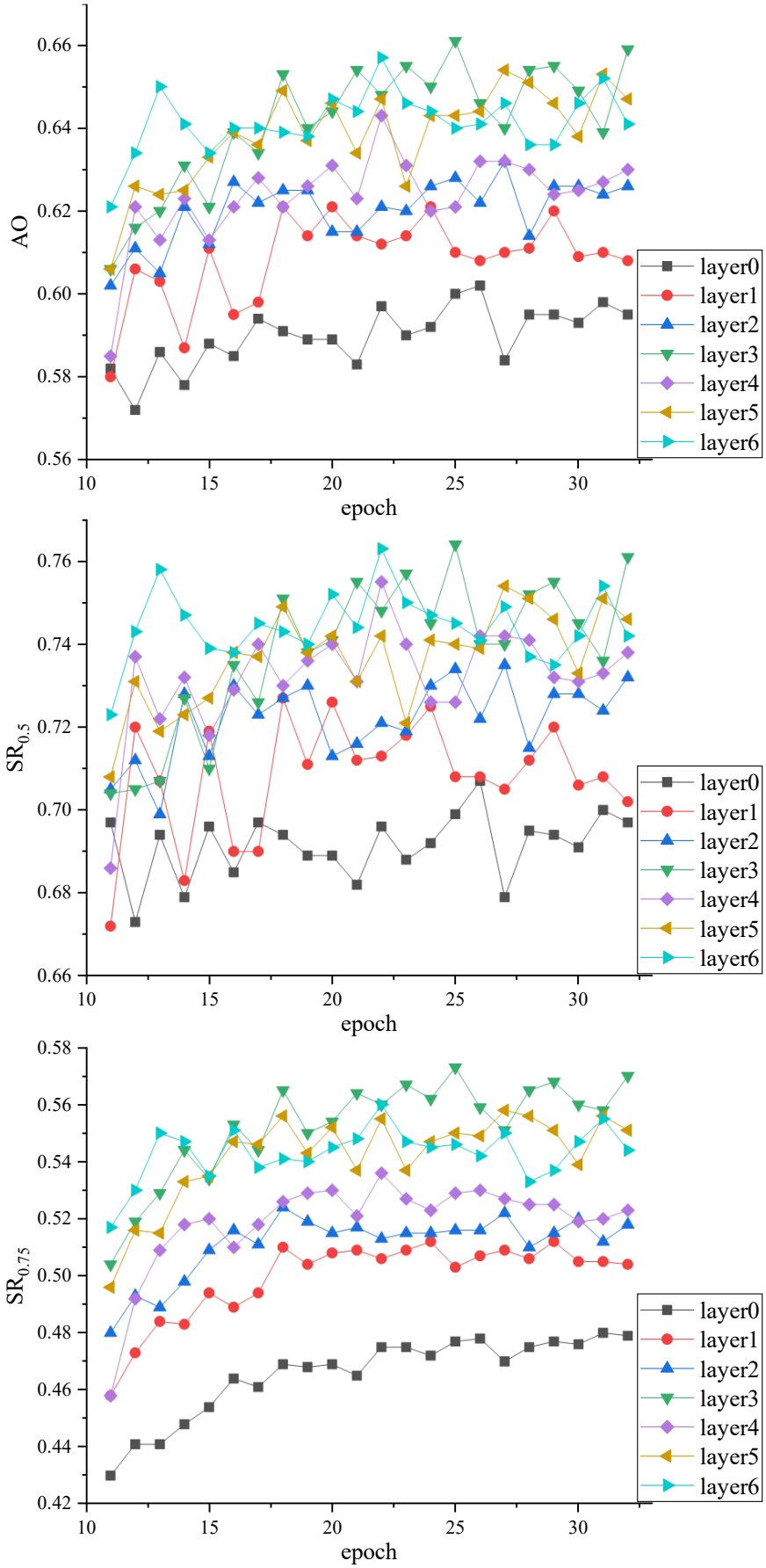

图 33　SiamWAVE 的七种配置在 GOT-10k 测试集的逐轮指标

该对比实验表明，简单地堆叠一至二层基于多层感知机的相关运算模块并不能取得相对于预相关运算模块的显著的性能提升；当层数继续增多至 3 层时，基于多层感知机的相关运算模块表现最优；在该数据集上层数继续增多并不一定能带来性能的提升。

3.4.3 LaSOT 测试集实验结果

(1) 算法评比

本章设计的基于相位感知注意力空间特征融合的孪生网络跟踪算法在 LaSOT 数据集上与近年各大顶级会议上发表的跟踪算法进行指标评比结果如表 5 与图 34 所示，其中表 5 和图 34 均展示成功率和精确率两项指标，各指标的解释详见 2.3 节。

表 5 SiamWAVE 与各跟踪算法在 LaSOT 测试集的指标评比结果

跟踪算法名称	LaSOT 测试集	
	成功率（%）	精确率（%）
AiATrack[43]	**69.0**	**73.8**
TransT[20]	64.9	69
SAOT[42]	61.6	70.8
PrDimp50[44]	59.8	-
AutoMatch[38]	58.3	59.9
DiMP[47]	56.9	-
Ocean[46]	56.0	56.6
SiamWAVE(layer6)	**55.2**	**55.6**
SiamWAVE(layer4)	**55.2**	**55.4**
SiamWAVE(layer3)	**54.7**	**54.3**
SiamFC++[48]	54.4	-
SiamWAVE(layer5)	**53.5**	**52.7**
PGNet[15]	53.1	60.5
SiamGAT[16]	53.0	53.9
SiamWAVE(layer2)	**52.7**	**52.3**
SiamWAVE(layer1)	**51.6**	**50.6**
ATOM[50]	51.4	50.5
SiamRPN++[12]	49.6	56.9
ROAM[49]	44.7	44.5
SiamDW[11]	**38.4**	-
SPM[45]	-	-

SiamRPN[51]	-	-

从图表中的结果可以看出，本章算法在 LaSOT 数据集上的表现有一些竞争力：该算法在 layer6 配置情况下相较于基线算法 SiamDW[11]的成功率指标（表中蓝色标出）有了近 17%的效果提升，相较于其它算法也均在成功率与精确率上有了部分提升，这说明设计的相关运算模块在空间维度使用简单的多层感知机结构将模板图像特征和搜索图像特征进行融合也可以获得不输于卷积的性能表现。

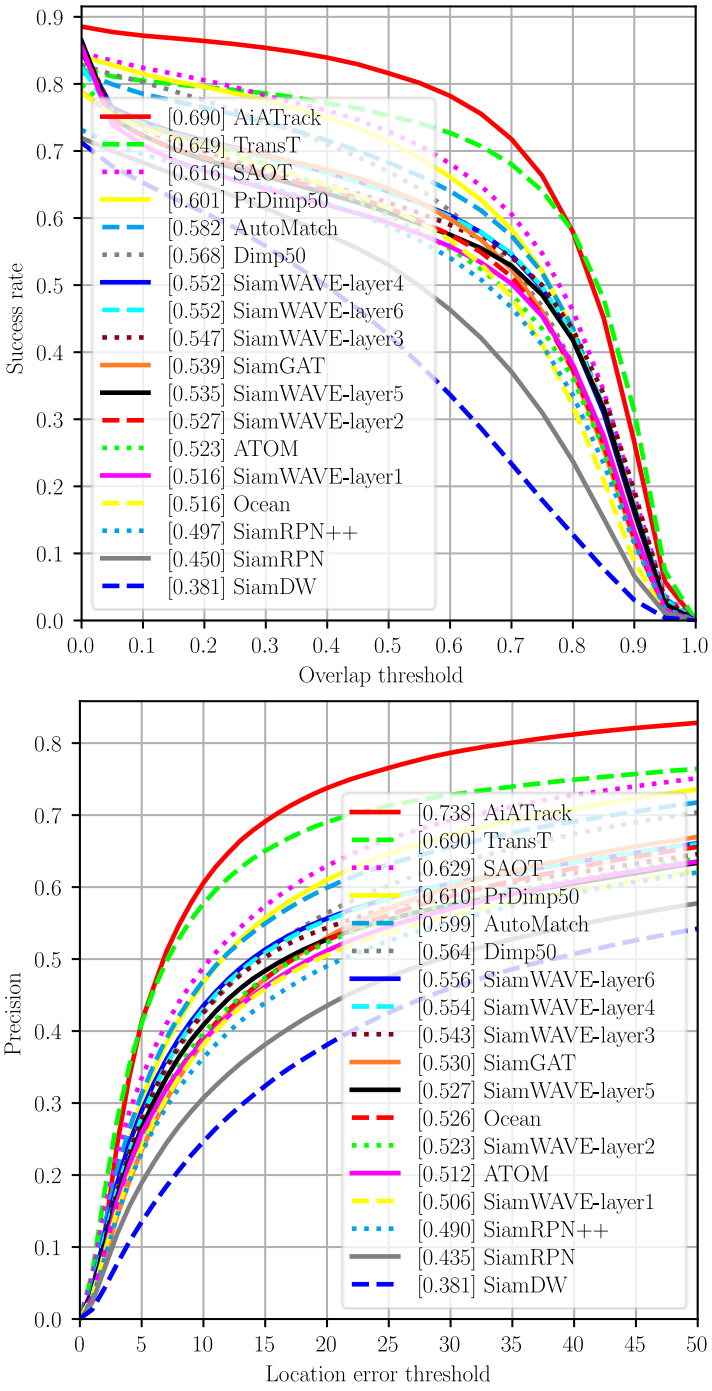

图 34　SiamWAVE 在 LaSOT 测试集的成功率图（上）与精确图（下）

然而，由于本章设计的算法只改进了孪生网络目标跟踪算法中的相关运算部分，其它部分的结构没有配适长时跟踪场景的需求，如添加模板更新操作等，因此它的表现略逊于含有模板更新操作的 Ocean[46]等算法。

(2) 对比实验

表 5 也展示了本章算法的六种配置的表现，随着设计的相关运算模块的逐层添加，其在 LaSOT 数据集上的性能表现也基本逐渐提升，这一定程度上表明了该相关运算模块的有效性。如图 35 所示，为了进一步证明设计的相关运算模块的有效性，本章设计了对比实验：包括只加预相关运算模块（即 layer0）在内的全部七种配置，在第 11 到第 32 轮训练过程中的各轮成功率指标与精确率指标进行对比。

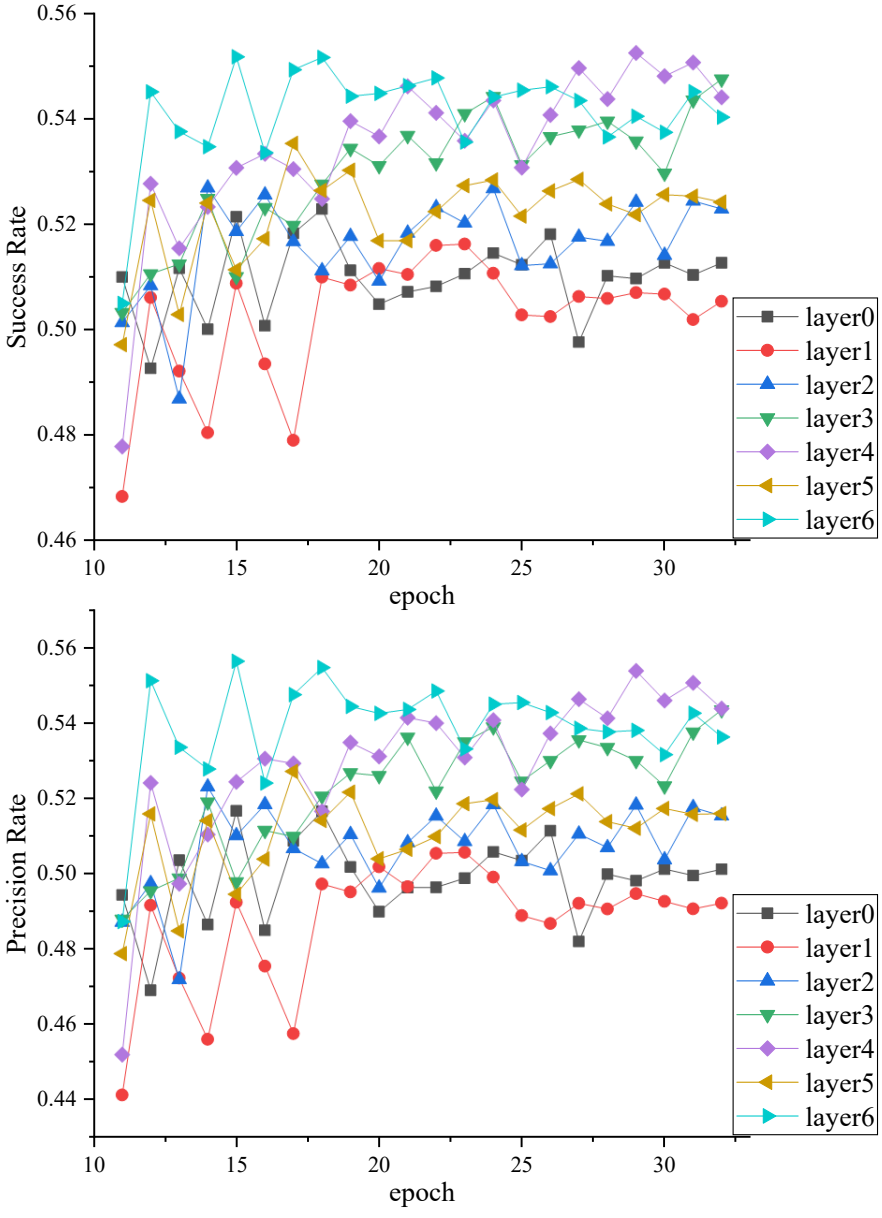

图 35 SiamWAVE 的七种配置在 LaSOT 测试集的逐轮指标

该对比实验表明，简单地堆叠一至二层基于多层感知机的相关运算模块并不能取得相对于预相关运算模块的显著的性能提升；当层数设置为 4 层或者 6 层时，基于多层感知机的相关运算模块表现最优；在该长时跟踪数据集上层数继续增多也并不一定能带来性能的提升。

3.5 本章小结

本章提出了一种基于相位感知注意力空间特征融合的孪生网络跟踪算法，该算法对孪生网络目标跟踪中的相关运算技术进行改进：设计了一种预相关运算模块与一种基于多层感知机的相关运算模块，其中后者将图像或图像特征视为一种“波”并采用相位感知注意力机制进行空间维度的特征融合。

在实验部分，本章将上述两种模块组合设置了七种不同的配置，并在 OTB100、GOT-10k 和 LaSOT 三个数据集上进行测试与对比。根据实验结果，本章设计的算法相较于基线算法都有了较高的性能提升，相较于大部分基于卷积的相关运算算法有更好的表现，从实验的角度证明了这种多层感知机的相关运算模块及其相位感知注意力机制的有效性。本章采用的简单的多层感知机结构以及不含显式注意力算子（查询算子、键算子和值算子）的计算使得本章对于相关运算关键技术的改进具有较小的计算开销，能够在第六章进行硬件实现。

第四章 基于图注意力通道特征融合的孪生网络跟踪算法

前文所述的相关运算方式在通道维度简单地使用线性结构进行特征融合，而现有的特征在通道维度进行相关运算的方式鲜有探究，主要原因是缺乏有效的通道特征相关性的建模，以及缺乏有效根据通道特征相关性的相关运算。针对这些问题，本章提出了一种基于图注意力的通道特征融合算法，使用 Watts-Strogatz 算法预先生成图结构建模通道特征相关性，再使用图注意力机制根据图结构进行相关运算。

4.1 引言

图结构对拓扑数据的表征及图神经网络对图结构的学习已经广泛应用于各种领域，如社交网络推荐[52]、3D 点云分割[53]、分子性质预测[54]和多智能体强化学习[55]等。在计算机视觉领域，最近的研究[56]表明图结构的建模和学习，对于建模图像块之间的相关性以及图像数据的特征提取是有意义的。

表 6 列举了最近采用图结构建模图像块之间相关性并使用图神经网络进行学习的特征提取网络在 ImageNet 数据集上完成图像分类任务的相关性能，其分类准确率指标已经超过了很多传统的深度卷积神经网络、多层感知机网络和视觉 Transformer 网络。

表 6 近年视觉图神经网络在 ImageNet 分类任务上的表现

模型	参数量	FLOPs	Top-1 acc. (%)
ViG-Ti[56]	7.1M	1.3B	73.9
ViG-S[56]	22.7M	4.5B	80.4
ViG-B[56]	86.8M	17.7B	82.3
Pyramid ViG-Ti[56]	10.7M	1.7B	78.2
Pyramid ViG-S[56]	27.3M	4.6B	82.1
Pyramid ViG-M[56]	51.7M	8.9B	83.1
Pyramid ViG-B[56]	92.6M	16.8B	83.7

图结构中节点之间边的连接将直接影响节点之间的相关性。如图 36 左图所示，节点 1 和节点 2 之间有边的存在，因此这两个节点之间存在直接相关性；节点 1 和节点 3 之间没有边的存在，但是因为节点 1 和节点 4 之间有边的存在并且节点 4 和节点 3 之间有边的存在，所以节点 1 和节点 3 之间存在间接相关性。有边存在的节点是邻居节点，如节点 2 是节点 1 的邻居节点，节点 3 不是节点 1 的邻居节点。

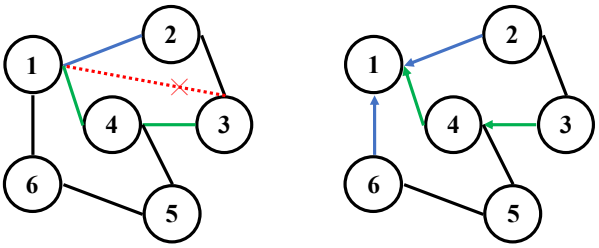

图 36　图结构及图上的消息传播机制

学习处理图结构上的图数据往往采用图神经网络，该类网络的核心思想之一是图上的消息传播机制。现有的图神经网络设计了各种图卷积算子，图卷积操作的每一次作用，将会以一种方式从各个节点的邻居节点收集它们的信息，这就是消息传播机制。如图 36 右图，一次图卷积操作将会使得节点 1 从它的邻居节点 2、4、6 上收集信息，因此节点 3 的信息可以通过两次图卷积操作传播到节点 1。

图结构建立以后，图数据之间的直接或间接相关性可以被图神经网络捕捉并学习到，因此图结构可以用来建模复杂抽象的数据之间的相关性，图神经网络可以用来学习图结构上的数据。

本章目标是基于图神经网络设计出一个能够有效融合模板图像特征和搜索图像特征两种图像特征张量的相关运算。首先，采用一些简单操作将模板图像特征和搜索图像特征进行预融合，设计预相关运算网络，得到两种特征初步融合的结果，本章设计的预相关运算模块与第三章的预相关运算模块一致，因此本章不再赘述。接着介绍 Watts-Strogatz 随机图生成算法[错误!未找到引用源。]，并从数学上证明使用这种固定结构的静态图可以满足动态数据的学习。然后介绍基于图神经网络的相关运算模块，采用图注意力机制对图上数据进行学习，完成模板图像特征和搜索图像特征的相关运算。最后，设计的基于图注意力通道特征融合的孪生网络跟踪算法将在 GOT-10k、YT-BB、COCO、DET 和 VID 五个数据集的训练集上进行训练，在 OTB100、GOT-10k 和 LaSOT 三个数据集的测试集上进行测试。

4.2 图结构生成模块

图结构生成模块的作用是产生一个图结构，用以建模预相关运算模块输出的特征张量通道特征之间的相关性。预相关运算模块在 3.2 节已经详细阐述，最后一个仿射变换操作的输出特征$\boldsymbol{s}_3 \in \mathbb{R}^{b\times c\times H_x\times W_x}$是整个预相关运算模块的输出，图结构生成模块需要生

成节点数为通道维度c的静态图。

本节采用 Watts-Strogatz 随机图生成算法[57]，它是一种 Erdos-Renyi 随机图生成算法[58]的改进算法，其生成的图结构具有较短的平均路径长度和较高的聚类性。

4.2.1 Erdos-Renyi 随机图生成算法

这是一种生成随机无向图最简单和常用的方法，生成的图结构称之为 ER 图，算法包括以下两种紧密相关的变体：

①G_{np}拥有n个节点，且边(u,v)以独立同分布的概率p产生的无向图。具体生成方法是按照某个次序考虑$\binom{n}{2}$条可能边中的每一条，然后以概率p独立地往图上添加每条边，其中$\binom{\cdot}{\cdot}$是组合数运算符。

②G_{nm}拥有n个节点，且其中m条边按照均匀分布采样生成的无向图。具体生成方法是均匀选取$\binom{n}{2}$条可能边中的一条并且将其添加为图的边，然后再独立且均匀地随机选取剩余$\binom{n}{2}-1$条可能边中的一条并且将其添加为图的边，直到选择了m条边为止。

然而，ER 图并不具有现实生活中的图的重要属性[59]，Watts-Strogatz 随机图生成算法对它进行了改进。

4.2.2 Watts-Strogatz 随机图生成算法

该模型能够部分解释许多现实生活中的图中的“小世界”现象，比如芽殖酵母脂肪代谢的信息交流[60]等。假设生成一个节点数为N的随机图，节点记为$n_0, n_1, \cdots, n_{N-1}$，该算法还需要被提供平均度$K$（假定为偶数）和一个参数$\beta$，其中$0 \le \beta \le 1$，且有$1 \ll \ln N \ll K \ll N$。以图 37 为例，随机图的生成包括以下两步：

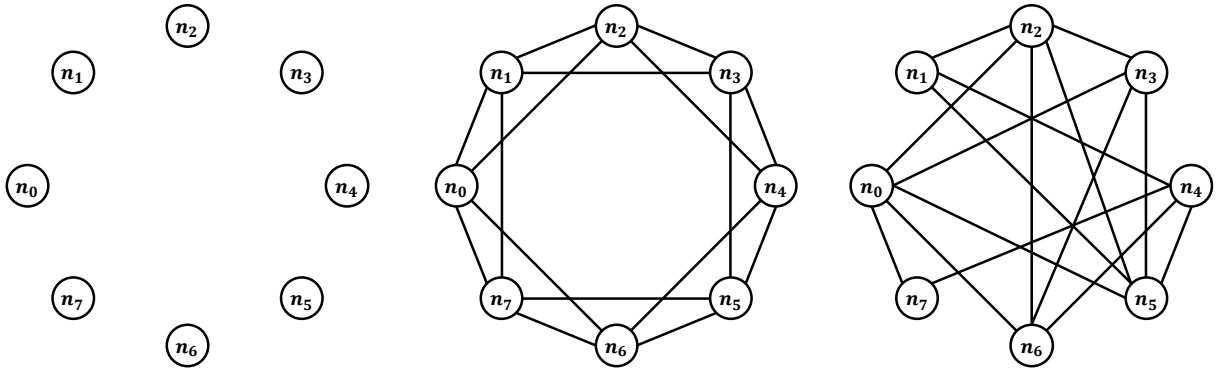

图 37 Watts-Strogatz 随机图生成示例

①构建一个正则环点阵。该图有N个节点，每个节点和 K 个相邻的节点相连，其中

每一侧有$K/2$个，即当且仅当满足下述条件时存在边(n_i, n_j)：

$$0 < |i-j| \bmod \left(N-1-\frac{K}{2}\right) \leq \frac{K}{2} \tag{4.1}$$

其中· mod ·是取模运算符。

如图 37 左图所示，以$N=8, K=4$为例，$N-1-\frac{K}{2}=5$，则：

对于n_0和n_1有$0<|0-1| \bmod 5=1\leq 2$，存在边；

对于n_0和n_2有$0<|0-2| \bmod 5=2\leq 2$，存在边；

对于n_0和n_3有$0<|0-3| \bmod 5=3>2$，不存在边；

对于n_0和n_4有$0<|0-4| \bmod 5=4>2$，不存在边；

对于n_0和n_5有$|0-5| \bmod 5=0$，不存在边；

对于n_0和n_6有$0<|0-6| \bmod 5=1\leq 2$，存在边；

对于n_0和n_7有$0<|0-7| \bmod 5=2\leq 2$，存在边；

其余同理，最终可以得到如图 37 中图的图结构。

②对N个节点中的每个节点，选出其与最右侧第$K/2$个相邻节点之间的边，即满足下述条件的所有的边$(n_i, n_{j \bmod N})$：

$$i < j \leq i + \frac{K}{2} \tag{4.2}$$

将这些边以概率β重新连接，即将$(n_i, n_{j \bmod N})$替换成(n_i, n_k)。

其中n_k以等概率从所有可能的节点中选取，并且避免出现自回路（self-loops）和重复连接（link duplication），自回路是指新选取的k等于i，重复连接是指新选取的k等于$j \bmod N$。

如图 37 中图所示，以$N=8, K=4, \beta=0.5$为例，$\frac{K}{2}=2$，则：

对于边(n_0, n_1)，选择重新连接，选取建立新边(n_0, n_5)；

对于边(n_0, n_2)，选择不重新连接；

对于边(n_1, n_2)，选择不重新连接；

对于边(n_1, n_3)，选择重新连接，选取建立新边(n_1, n_4)；

对于边(n_7, n_0)，选择不重新连接；

对于边(n_7, n_1)，选择重新连接，选取建立新边(n_7, n_4)；

其余同理，最终可以得到如图 37 右图的图结构，至此 Watts-Strogatz 随机图生成算法结束。

如图 38 所示，本章内容预先生成了多张节点数N为 128，平均度K为 4，参数β为 1 的 Watts-Strogatz 随机图，并选用左上角的随机图进行本章的实验。

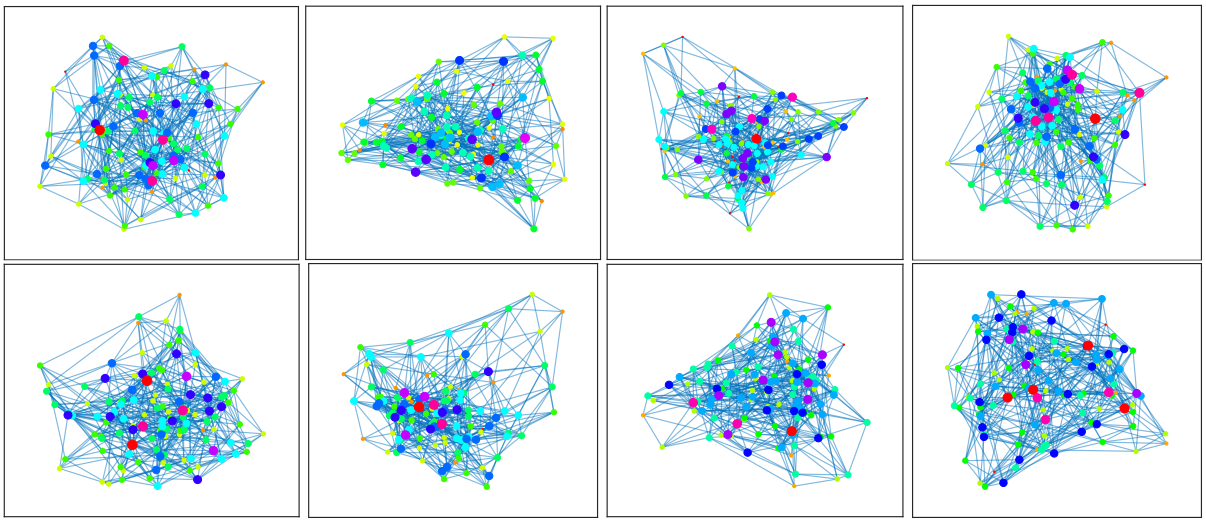

图 38 预先生成的若干张 Watts-Strogatz 随机图示例

4.2.3 静态图可行性证明

在实际进行视频单目标跟踪时，每一帧的相关运算都采用 Watts-Strogatz 随机图生成算法动态生成图结构是不现实的，这会产生额外的计算开销，严重影响跟踪算法的实时性。为了解决这一问题，本文采用预先生成的 Watts-Strogatz 随机图，并在实时跟踪的时候不再改变图的结构。下面将从理论上证明 Watts-Strogatz 静态图的可行性。

首先，图上各个节点的度中心性直接影响了图的属性，其中一个节点的度中心性被定义为连接该节点的边的数量。

假设使用 Watts-Strogatz 随机图生成算法生成若干个节点数均为n的随机图，记这些随机图为：

$$\mathcal{G}_1, \mathcal{G}_2, \cdots, \mathcal{G}_i, \cdots$$

记第i个随机图的n个节点的度为：

$$\{d_j^i\}, \quad j = 1,2, \cdots, n$$

记第i个随机图的平均度为：

$$M^i = \frac{1}{n}\sum_{j=1}^{n} d_j^i \tag{4.3}$$

则有下述定理：

定理 1：若$n \to \infty$，则$M^i \to K$，其中K是一个常数。

证明：已知图上节点的度服从正态分布，即$d^i \sim \mathcal{N}(\mu^i, \sigma^i)$，则由大数定律可知：

当$n \to \infty$时，样本均值$\frac{1}{n}\sum_{j=1}^{n} d_j^i$趋向于随机变量的期望，这里正态分布的期望就是$\mu^i$，而 Watts-Strogatz 随机图的度的期望是一个常数K，故得证。

因此当 Watts-Strogatz 随机图的节点数较大时，可以近似认为 Watts-Strogatz 算法动态生成的随机图整体上性质没有较大的差异，可以用来对通道特征的相关性进行稳定地建模。

4.3 图神经网络相关运算模块

图神经网络相关运算模块的基本处理流程如图 39 所示，在该相关运算模块中，使用预先生成的 Watts-Strogatz 随机图建模通道特征之间地相关性，特征张量的每一个通道都被看作是图上的一个节点（node），相关运算建立在节点聚合操作之中：图注意力机制动态学习邻居节点的权重并自适应地加权求和，使得节点特征进行聚合，即通道特征进行融合。

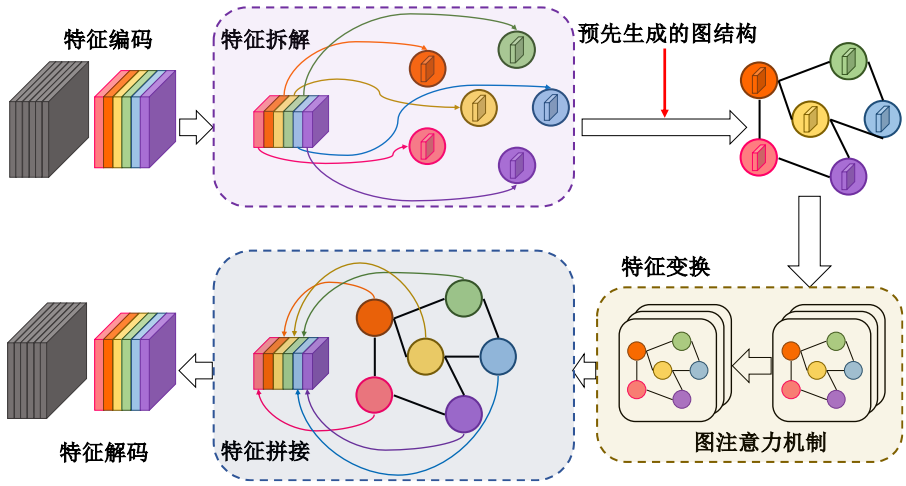

图 39　图神经网络相关运算模块

4.3.1 图注意力机制

图上的节点聚合操作基于消息传播机制，每一次节点聚合操作都会将邻居节点的信息按照一定的方式聚合到该节点中，图神经网络利用这种节点聚合操作完成对图的学习。在图神经网络中，节点上的信息是该节点的特征，进行一次节点聚合操作便是进行了一次图卷积运算，不同的节点聚合方式产生了不同的图卷积运算。本文所采用的是基于图

注意力机制的图卷积运算——图注意力卷积[17]，它依赖注意力机制动态计算邻居节点的不同权重，并利用这些权重和节点特征进行加权求和。

记 4.2 节预先生成的图为$\mathcal{G}$：

$$\mathcal{G} = (\mathcal{V}, \mathcal{E})$$

其中$\mathcal{V}$是图$\mathcal{G}$上所有节点的集合，$\mathcal{E}$是图$\mathcal{G}$上所有边的集合。对于每一个$(i,j) \in \mathcal{E}$，令$\boldsymbol{e}_{ij}$表示节点i和节点j之间的相关得分：

$$\boldsymbol{e}_{ij} = f(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{4.4}$$

其中，$\boldsymbol{x}_i, \boldsymbol{x}_j$分别是节点$i$和节点$j$的特征。

本文使用特征之间的内积作为相似性度量。为了自适应地学习节点之间更好的表示，先对节点特征进行线性变换，然后对变换后的特征取内积来计算相关得分：

$$f(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{W}_q \boldsymbol{x}_i)^{\mathrm{T}} (\boldsymbol{W}_k \boldsymbol{x}_j) \tag{4.5}$$

其中$\boldsymbol{W}_q$和$\boldsymbol{W}_k$是线性变换矩阵，矩阵中的元素为待训练的参数。

使用 Softmax 函数将$\boldsymbol{e}_{ij}$正则化：

$$\boldsymbol{a}_{ij} = \frac{\exp(\boldsymbol{e}_{ij})}{\sum_{k \in \mathrm{Neighbour}(i)} \exp(\boldsymbol{e}_{ik})} \tag{4.6}$$

其中$\mathrm{Neighbour}(i)$是节点i的邻居节点集合。

$\boldsymbol{a}_{ij}$衡量了图$\mathcal{G}$上节点j向节点i传递的信息的重要性，可以作为节点聚合操作时节点j的特征的权重。由于该权重是动态学习线性变换矩阵$\boldsymbol{W}_q$和$\boldsymbol{W}_k$得到的，因此可以自适应调整，这便是图上的注意力机制，其中$\boldsymbol{a}_{ij}$是注意力权重。

节点i的节点聚合操作可以表示为：

$$\boldsymbol{v}_i = \sum_{j \in \mathrm{Neighbour}(i)} \boldsymbol{a}_{ij} \boldsymbol{W}_v \boldsymbol{x}_j \tag{4.7}$$

其中$\boldsymbol{W}_v$是参数可学习的线性变换矩阵。

最后将节点i经过节点聚合操作从邻居节点聚合得到的特征$\boldsymbol{v}_i$与节点i的原始特征$\boldsymbol{x}_i$融合，得到一个更强大的特征表示$\hat{\boldsymbol{x}}_i$：

$$\hat{\boldsymbol{x}}_i = \mathrm{Softmax}\Big(\mathrm{ReLU}\big(\boldsymbol{v}_i \,\|\, (\boldsymbol{W}_v \boldsymbol{x}_i)\big)\Big) \tag{4.8}$$

其中$\|$表示拼接（Concat）操作，ReLU是激活函数。

对于任意节点$i \in \mathcal{V}$计算$\hat{\boldsymbol{x}}_i$，最终完成一次图注意力卷积运算。

4.3.2 相关运算模块

基于图注意力机制的图神经网络相关运算模块的功能是将预相关运算模块的初步相关运算结果再进行通道维度的深度相关运算。它接收 3.2 节预相关运算模块的输出，即仿射变换特征$\boldsymbol{s}_3 \in \mathbb{R}^{b\times c\times H_x\times W_x}$，输出深度相关运算结果$\boldsymbol{o} \in \mathbb{R}^{b\times c\times H_x\times W_x}$。

由于该模块中图注意力机制处理的对象是通道特征，因此需要将通道特征进行编码，再将编码后的特征按通道维度进行拆解，拆解后的特征可以作为预先生成的 Watts-Strogatz 随机图上的节点特征，在经过图注意力机制的节点聚合操作以后，通道特征就已经完成融合，此时需要重新按通道维度拼接通道特征，再将通道特征进行解码得到相关运算的结果。

记本节相关运算模块的输入是$\boldsymbol{x}_{in} = \boldsymbol{s}_3 \in \mathbb{R}^{b\times c\times H_x\times W_x}$，最后深度相关运算结果$\boldsymbol{o}$作为基于图注意力机制的图神经网络相关运算模块的输出$\boldsymbol{x}_{out} \in \mathbb{R}^{b\times c\times H_x\times W_x}$。

如图 39 所示，基于图注意力机制的图神经网络相关运算模块具体分为下述 5 个步骤进行：

(1) 特征编码：

将$\boldsymbol{x}_{in}$的通道特征进行编码，可以生成图上的节点特征：

先将$\boldsymbol{x}_{in}$进行变形操作，变换为$\boldsymbol{x}'_{in} \in \mathbb{R}^{b\times H_x\times W_x\times c\times 1}$，然后使用两个参数可学习的线性变换矩阵$\boldsymbol{W}^{in}$和$\boldsymbol{b}^{in}$进行特征编码：

$$\boldsymbol{x}^f = \left(\boldsymbol{W}^{in}\right)\cdot_b \left(\boldsymbol{x}'_{in}\right) +_b \left(\boldsymbol{b}^{in}\right) \tag{4.9}$$

其中，$\cdot_b$为广播乘积，$+_b$为广播求和。

参数矩阵$\boldsymbol{W}^{in} \in \mathbb{R}^{1\times c'}$，因此$\left(\boldsymbol{W}^{in}\right)\cdot_b \left(\boldsymbol{x}'_{in}\right) \in \mathbb{R}^{b\times H_x\times W_x\times c\times c'}$；参数矩阵$\boldsymbol{b}^{in} \in \mathbb{R}^{c'}$，因此$\boldsymbol{x}^f \in \mathbb{R}^{b\times H_x\times W_x\times c\times c'}$，是一个五维张量。

其中c是通道维度，c'是通道特征的维度。因此预先生成的 Watts-Strogatz 随机图具有c个节点，节点特征的维度为c'或$b \times H_x \times W_x \times c'$。

(2) 特征拆解：

将$\boldsymbol{x}^f$按通道维度进行拆解：

$$\boldsymbol{x}^f = \left\{\boldsymbol{x}_1^f, \boldsymbol{x}_2^f, \cdots, \boldsymbol{x}_c^f\right\},\ \ \boldsymbol{x}_j^f \in \mathbb{R}^{b\times H_x\times W_x\times c'},\ \ j = 1,2,\cdots,c$$

拆解得到的特征$\boldsymbol{x}_j^f$按序作为 Watts-Strogatz 随机图的节点特征$\boldsymbol{x}_i$，进行赋值的过程

如图 39 的紫色虚线框内所示：

$$\begin{cases} \boldsymbol{x}_1 \leftarrow \boldsymbol{x}_1^f \\ \boldsymbol{x}_2 \leftarrow \boldsymbol{x}_2^f \\ \quad \vdots \\ \boldsymbol{x}_c \leftarrow \boldsymbol{x}_c^f \end{cases} \tag{4.10}$$

其中←是赋值操作。

(3) 特征变换：

采用 4.3.1 节中的图注意力机制进行一次图注意力卷积运算，使得通道特征进行融合，过程如图 39 的黄色虚线框内所示，最终得到特征变换的输出：

$$\hat{\boldsymbol{x}} = \{\hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2, \cdots, \hat{\boldsymbol{x}}_c\},\ \ \hat{\boldsymbol{x}}_i \in \mathbb{R}^{b \times H_x \times W_x \times c'},\ \ i = 1,2,\cdots,c$$

(4) 特征拼接：

将图上的节点特征取出，赋值给新的通道特征$\hat{\boldsymbol{x}}^f$，该过程如图 39 的蓝色虚线框内所示：

$$\begin{cases} \hat{\boldsymbol{x}}_1^f \leftarrow \hat{\boldsymbol{x}}_1 \\ \hat{\boldsymbol{x}}_2^f \leftarrow \hat{\boldsymbol{x}}_2 \\ \quad \vdots \\ \hat{\boldsymbol{x}}_c^f \leftarrow \hat{\boldsymbol{x}}_c \end{cases} \tag{4.11}$$

其中$\hat{\boldsymbol{x}}^f = \{\hat{\boldsymbol{x}}_1^f, \hat{\boldsymbol{x}}_2^f, \cdots, \hat{\boldsymbol{x}}_c^f\}$，$\hat{\boldsymbol{x}}_j^f \in \mathbb{R}^{b \times H_x \times W_x \times c'}$，$j = 1,2,\cdots,c$。

按通道维度原来的顺序进行拼接得到通道特征融合后的结果：

$$\hat{\boldsymbol{x}}^f \in \mathbb{R}^{b \times H_x \times W_x \times c \times c'}$$

这是一个五维张量。

(5) 特征解码：

将$\hat{\boldsymbol{x}}^f$的通道特征进行解码，得到相关运算的结果$\boldsymbol{o}$。

使用两个参数可学习的线性变换矩阵$\boldsymbol{W}^{out}$和$\boldsymbol{b}^{out}$进行特征解码：

$$\boldsymbol{o}' = (\boldsymbol{W}^{out}) \cdot_b (\hat{\boldsymbol{x}}^f) +_b (\boldsymbol{b}^{out}) \tag{4.12}$$

其中，$\cdot_b$为广播乘积运算，$+_b$为广播求和机制。

参数矩阵$\boldsymbol{W}^{out} \in \mathbb{R}^{c' \times 1}$，因此$(\boldsymbol{W}^{out}) \cdot_b (\hat{\boldsymbol{x}}^f) \in \mathbb{R}^{b \times H_x \times W_x \times c \times 1}$；参数矩阵$\boldsymbol{b}^{out} \in \mathbb{R}^{c}$，因此$\boldsymbol{o}' \in \mathbb{R}^{b \times H_x \times W_x \times c}$，是一个四维张量。

最后将$\boldsymbol{o}'$进行变形操作得到相关运算结果$\boldsymbol{o} \in \mathbb{R}^{b \times c \times H_x \times W_x}$。

4.4 实验与分析

为了验证本章的孪生网络目标跟踪算法（称为“SiamGATPlus”）以及设计的相关运算模块的有效性，实验分为表 7 中四种模型配置：

表 7　四种模型配置

模型配置	layer0	layer1	layer2	layer3

其中，layer0 是指仅使用预相关运算模块进行相关运算，layer1 到 layer3 是指在预相关运算模块的基础上堆叠对应数量的图神经网络相关运算模块。

四种配置的模型在全部 GOT-10k、YT-BB、COCO、DET 和 VID 五个训练集上进行模型训练，一共训练 32 轮（epoch），在前 10 轮冻结骨干网络使其参数在训练过程中不变，从第 11 轮开始对骨干网络和相关运算模块一并训练。四种配置的模型在 OTB100、GOT-10k 和 LaSOT 三个测试集上进行性能评估。

4.4.1 OTB100 测试集实验结果

(1) 算法评比

本章设计的基于图注意力通道特征融合的孪生网络跟踪算法在 OTB100 数据集上与近年各大顶级会议上发表的跟踪算法进行指标评比结果如表 8 与图 40 所示。

表 8　SiamGATPlus 与各跟踪算法在 OTB100 测试集的指标评比结果

跟踪算法名称	OTB100 测试集	
	成功率（%）	精确率（%）
SAOT[42]	**71.4**	**92.6**
AutoMatch[38]	**71.4**	**92.6**
SiamGAT[16]	70.9	91.6
SiamGATPlus(layer3)	**70.3**	**91.4**
SiamGATPlus(layer1)	**69.9**	**91.9**
SiamRPN++[12]	69.6	92.3
AiATrack[43]	69.6	91.7
PrDimp50[44]	69.5	89.7
TransT[20]	69.1	89.3
PGNet[15]	69.1	89.2
SPM[45]	68.7	89.9
Ocean[46]	68.4	92.0

DiMP[47]	68.4	-
SiamFC++[48]	68.3	-
ROAM[49]	68.1	90.8
SiamDW[11]	**67.4**	-
ATOM[50]	66.7	87.9
SiamGATPlus(layer2)	**66.6**	**86.3**
SiamRPN[51]	63.7	85.1

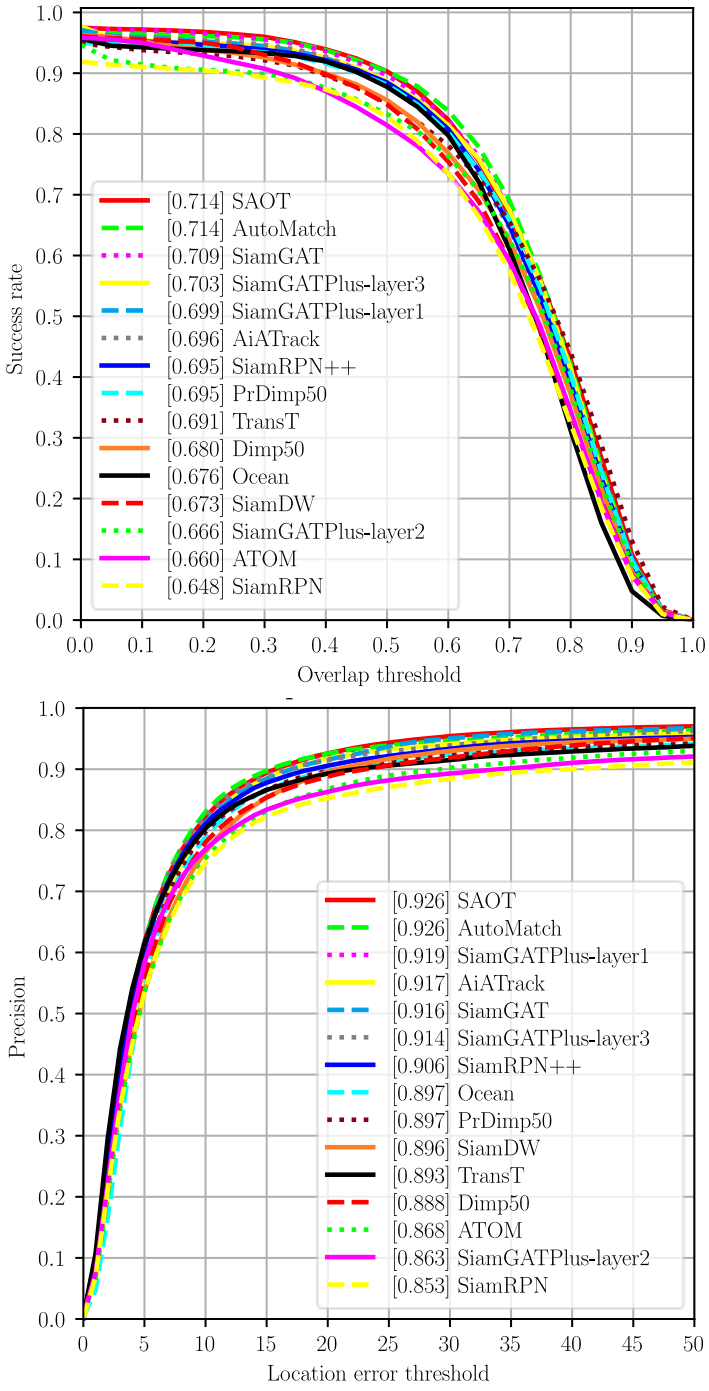

图 40 SiamGATPlus 在 OTB100 测试集的成功率图（上）与精确图（下）

从图表中的结果可以看出，本章算法在 OTB100 数据集上的表现具有竞争力：该算法在 layer3 配置情况下相较于基线算法 SiamDW[11]的成功率指标（表中蓝色标出）有了近 3%的效果提升，相较于其它算法也均在成功率与精确率上有了部分提升，这说明图结构的引入可以有效建模通道特征的相关性，基于图神经网络的相关运算模块有效进行了图结构上的相关运算。本章的算法在该数据集上与在空间维度使用图结构进行相关运算的算法 SiamGAT[16]指标相近，这说明重点进行通道维度的相关运算可以得到与进行空间维度相关运算持平的性能表现。

(2) 对比实验

图 41 在对比实验中展示了 SiamGATPlus 算法的四种配置之间的差异。

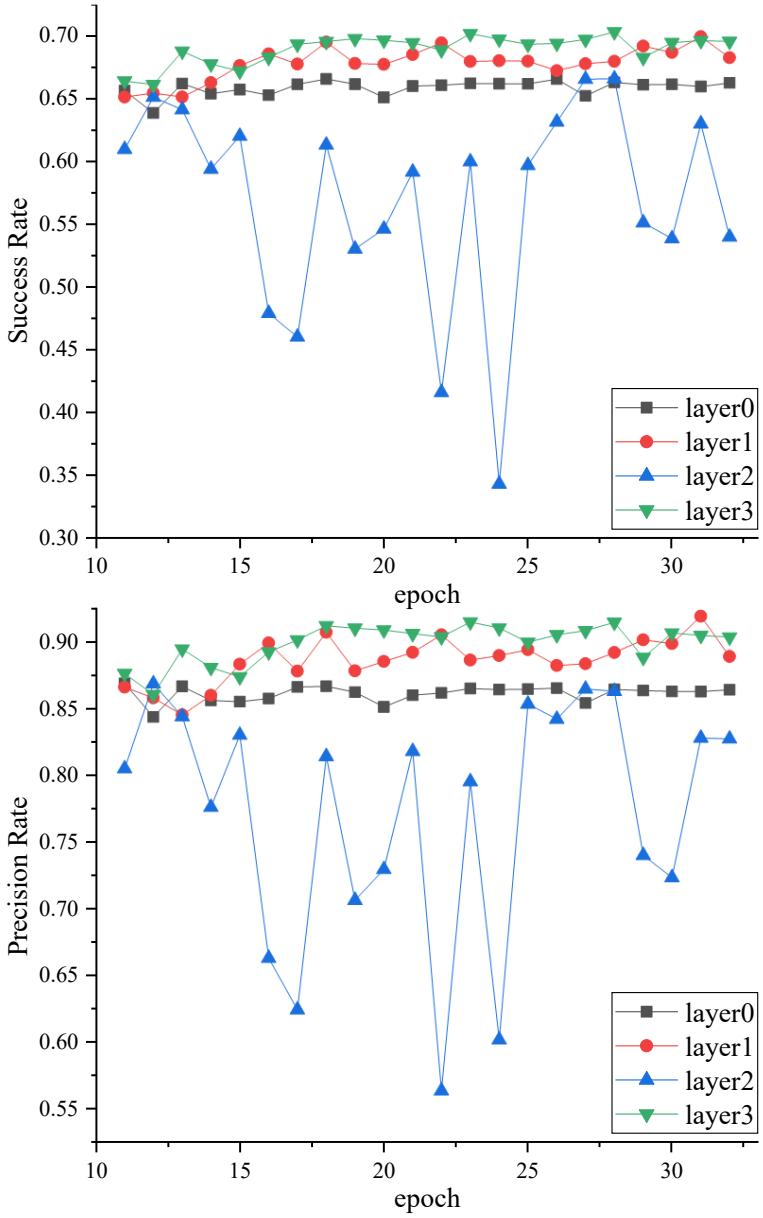

图 41 SiamGATPlus 的四种配置在 OTB100 测试集的逐轮指标

表 8 也展示了本章算法的三种配置的表现，随着设计的相关运算模块的逐层添加，其在 OTB100 数据集上的性能表现整体上逐渐提升，这一定程度上表明了该相关运算模块的有效性。如图 41 所示，为了进一步证明设计的相关运算模块的有效性，本章设计了对比实验：包括只加预相关运算模块（即 layer0）在内的全部四种配置，在第 11 到第 32 轮训练过程中的各轮成功率指标与精确率指标进行对比。

该对比实验表明，由于随机图的引入为模型的训练带来了一些不稳定性，简单堆叠一至二层基于图神经网络的相关运算模块的性能表现并不稳定；当层数继续增多至 3 层时，基于图神经网络的相关运算模块逐渐表现出性能提升效果。

4.4.2 GOT-10k 测试集实验结果

(1) 算法评比

本章设计的基于图注意力通道特征融合的孪生网络跟踪算法在 GOT-10k 数据集上与近年各大顶级会议上发表的跟踪算法进行指标评比结果如表 9 与图 42 所示，其中表 9 列出 AO、$SR_{0.5}$ 和 $SR_{0.75}$ 三项指标，图 42 单独绘制 GOT-10k 数据集上的成功率图，各指标的解释详见 2.3 节。

表 9 SiamGATPlus 与各跟踪算法在 GOT-10k 测试集的指标评比结果

跟踪算法名称	GOT-10k 测试集		
	AO（%）	$SR_{0.5}$（%）	$SR_{0.75}$（%）
AiATrack[43]	**69.6**	**80**	**63.2**
SPM[45]	68.7	89.9	
TransT[20]	67.1	76.8	60.9
SiamGATPlus(layer3)	**65.8**	**77.3**	**55.2**
AutoMatch[38]	65.2	76.6	54.3
SAOT[42]	64	74.9	-
PrDimp50[44]	63.4	73.8	54.3
SiamGAT[16]	62.7	74.3	48.8
SiamGATPlus(layer1)	**62.2**	**72.4**	**50.2**
SiamGATPlus(layer2)	**62.1**	**73.5**	**50.7**
Ocean[46]	61.1	72.1	-
DiMP[47]	61.1	71.7	49.2
SiamFC++[48]	59.5	69.5	47.9
ROAM[49]	46.5	53.2	23.6

SiamDW[11]	**41.6**	-	-
PGNet[15]	-	-	-
SiamRPN++[12]	-	-	-
ATOM[50]	-	-	-
SiamRPN[51]	-	-	-

从图表中的结果可以看出，本章算法在 GOT-10k 数据集上的表现具有很强的竞争力：该算法在 layer3 配置情况下相较于基线算法 SiamDW[11]的 AO 指标（表中蓝色标出）有了 24%的效果提升，相较于其它算法也均在 AO、$SR_{0.5}$ 和 $SR_{0.75}$ 三项指标上有了部分提升，这说明图结构的引入可以有效建模通道特征的相关性，基于图神经网络的相关运算模块有效进行了图结构上的相关运算。本章的算法在该数据集上指标超过了在空间维度使用图结构进行相关运算的算法 SiamGAT[16]，这说明重点进行通道维度的相关运算在一些场景下可以得到超过进行空间维度相关运算的跟踪算法的性能表现。

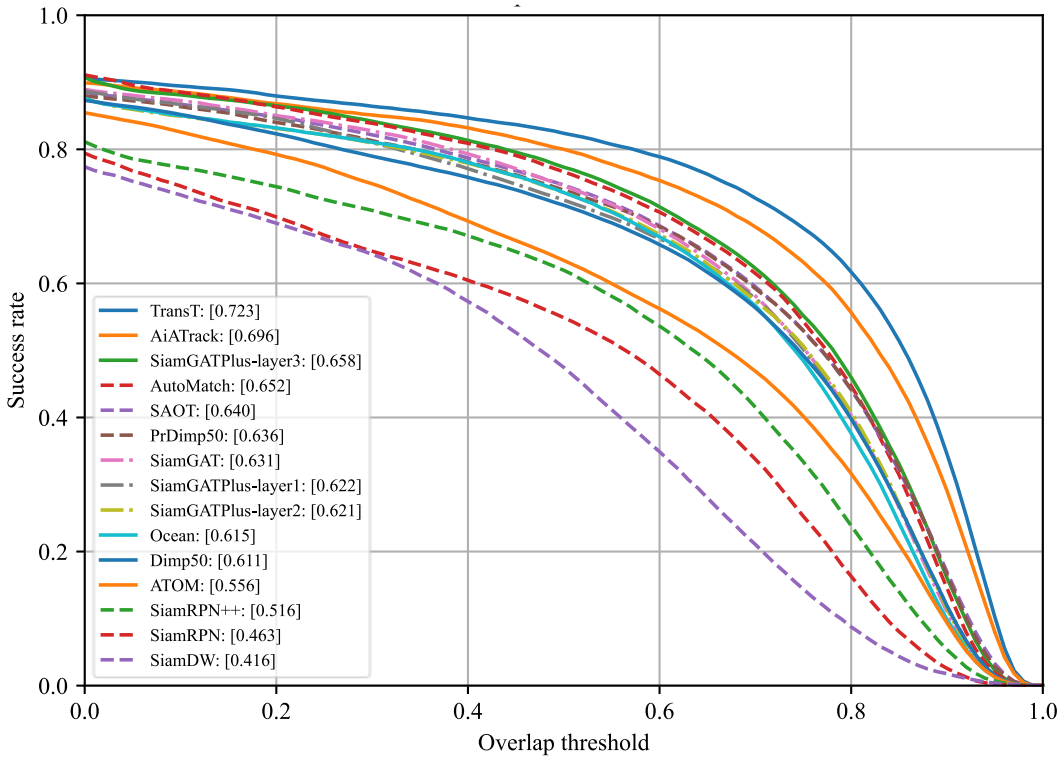

图 42　SiamGATPlus 在 GOT-10k 测试集的成功率图

(2) 对比实验

表 9 也展示了本章算法的三种配置的表现，随着设计的相关运算模块的逐层添加，其在 GOT-10k 数据集上的性能表现整体上逐渐提升，这一定程度上表明了该相关运算模块的有效性。图 43 进一步展示了 SiamGATPlus 算法的四种配置之间的差异。

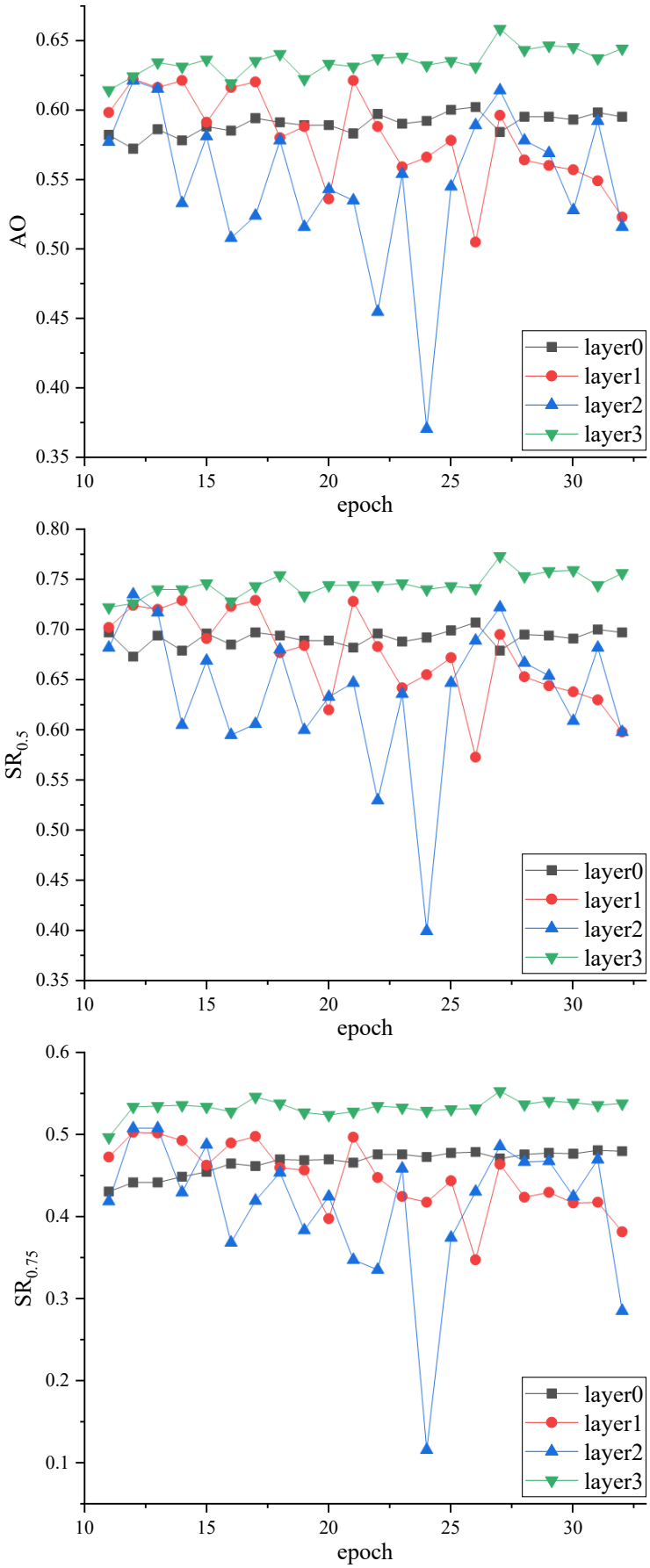

图 43 SiamGATPlus 的四种配置在 GOT-10k 测试集的逐轮指标

如图 43 所示，为了进一步证明设计的相关运算模块的有效性，本章设计了对比实验：包括只加预相关运算模块（即 layer0）在内的全部四种配置，在第 11 到第 32 轮训练过程中的各轮 AO、$SR_{0.5}$ 和 $SR_{0.75}$ 指标进行对比。

如 4.2.3 中证明所示，本实验中采用的节点数 128 并不一定使得证明满足大数定律的条件，因此本章采用的随机图为模型的训练带来了一些不稳定性。在该对比实验中，不添加图神经网络相关运算模块的 layer0 的逐轮训练结果较为稳定，但是添加图神经网络相关运算模块的 layer1 和 layer2 的训练结果存在较大的不稳定性，这表明该不稳定性确实是图的引入与学习带来的。当层数继续增多至 3 层时，基于图神经网络的相关运算模块逐渐表现出性能稳定提升效果。

4.4.3 LaSOT 测试集实验结果

(1) 算法评比

本章设计的基于图注意力通道特征融合的孪生网络跟踪算法在 LaSOT 数据集上与近年各大顶级会议上发表的跟踪算法进行指标评比结果如表 10 与图 44 所示，其中表 10 和图 44 均展示成功率和精确率两项指标，各指标的解释详见 2.3 节。

表 10　SiamGATPlus 与各跟踪算法在 LaSOT 测试集的指标评比结果

跟踪算法名称	LaSOT 测试集	
	成功率（%）	精确率（%）
AiATrack[43]	**69.0**	**73.8**
TransT[20]	64.9	69.0
SAOT[42]	61.6	70.8
PrDimp50[44]	59.8	-
AutoMatch[38]	58.3	59.9
DiMP[47]	56.9	-
Ocean[46]	56.0	56.6
SiamGATPlus(layer2)	**54.9**	**55.5**
SiamGATPlus (layer3)	**54.6**	**55.4**
SiamGATPlus(layer1)	**54.4**	**54.8**
SiamFC++[48]	54.4	-
PGNet[15]	53.1	60.5
SiamGAT[16]	53.0	53.9
ATOM[50]	51.4	50.5

SiamRPN++[12]	49.6	56.9
ROAM[49]	44.7	44.5
SiamDW[11]	**38.4**	-
SPM[45]	-	-
SiamRPN[51]	-	-

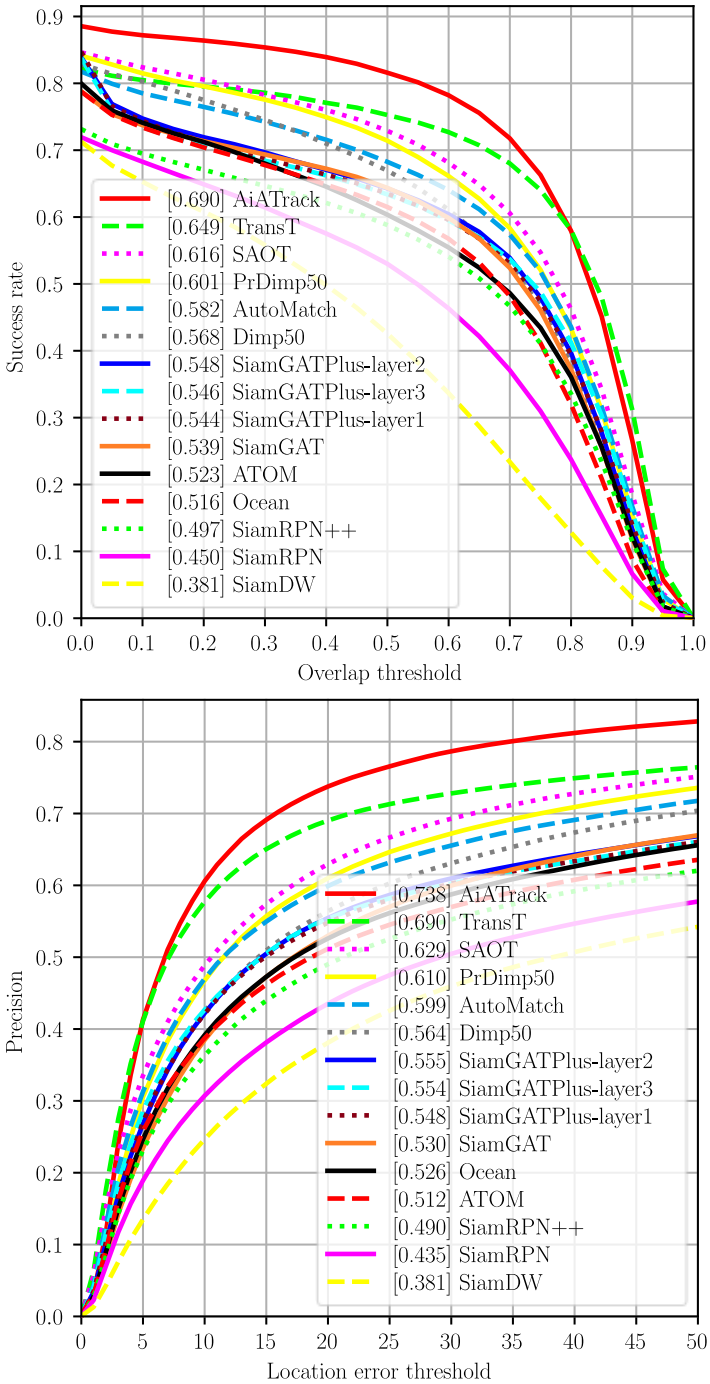

图 44 SiamGATPlus 在 LaSOT 测试集的成功率图（上）与精确图（下）

从图表中的结果可以看出，本章算法在 LaSOT 数据集上的表现有一些竞争力：该算法在 layer2 配置情况下相较于基线算法 SiamDW[11]的成功率指标（表中蓝色标出）有

了近 17%的效果提升，相较于其它算法也均在成功率与精确率上有了部分提升，这说明图结构的引入可以有效建模通道特征的相关性，基于图神经网络的相关运算模块有效进行了图结构上的相关运算。

本章的算法在该数据集上指标超过了在空间维度使用图结构进行相关运算的算法 SiamGAT[16]，这说明重点进行通道维度的相关运算在一些场景下可以得到超过进行空间维度相关运算的跟踪算法的性能表现。

然而，由于本章设计的算法只改进了孪生网络目标跟踪算法中的相关运算部分，其它部分的结构没有配适长时跟踪场景的需求，如添加模板更新操作等，因此它的表现略逊于含有模板更新操作的 Ocean[46]、DiMP[47]和 PrDimp[44]等算法。

由于本章设计的算法不能像 AutoMatch[38]一样根据特定目标及其场景自动选取合适的相关运算方式，难以适应长时跟踪场景中目标的复杂变化，如目标在搜索区域中短暂消失（out-of-view），因此在长时跟踪中的表现略逊于 AutoMatch 等算法。

本章设计的算法表现仍不及使用复杂沉重的 Transformer 结构设计相关运算模块的算法 TransT[20]、AiATrack[43]等，这表明在空间维度进行深度相关运算仍能取得十分有竞争力的跟踪性能表现，侧面表明了本章在通道维度进行相关运算的方式可以继续改进，即采用更深、更复杂、更大参数量的通道维度相关运算模块有可能取得更好的跟踪性能表现。

(2) 对比实验

表 10 也展示了本章算法的三种配置的表现，随着设计的相关运算模块的逐层添加，其在 LaSOT 数据集上的性能表现整体上逐渐提升，这一定程度上表明了该相关运算模块的有效性。如图 45 所示，为了进一步证明设计的相关运算模块的有效性，本章设计了对比实验：包括只加预相关运算模块（即 layer0）在内的全部四种配置，在第 11 到第 32 轮训练过程中的各轮成功率指标与精确率指标进行对比。

如 4.2.3 中证明所示，本实验中采用的节点数 128 并不一定使得证明满足大数定律的条件，因此本章采用的随机图为模型的训练带来了一些不稳定性。在该对比实验中，不添加图神经网络相关运算模块的 layer0 的逐轮训练结果较为稳定，但是添加图神经网络相关运算模块的 layer1 和 layer2 的训练结果存在较大的不稳定性，这表明该不稳定性确实是图的引入与学习带来的。当层数继续增多至 3 层时，基于图神经网络的相关运算

模块逐渐表现出性能稳定提升效果。

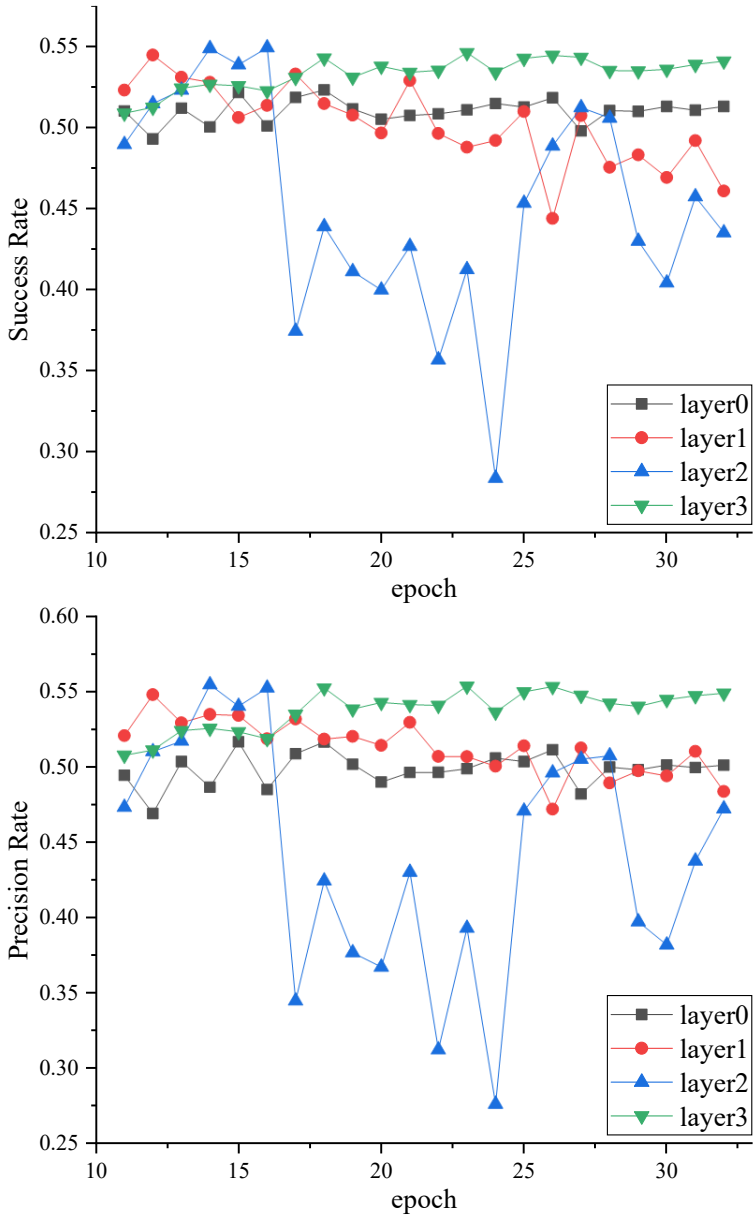

图 45 SiamGATPlus 的四种配置在 LaSOT 测试集的逐轮指标

4.5 本章小结

本章提出了一种基于图注意力通道特征融合的孪生网络跟踪算法，该算法对孪生网络目标跟踪中的相关运算技术进行改进：设计了一种预相关运算模块与一种基于图神经网络的相关运算模块，其中后者利用 Watts-Strogatz 随机图生成算法预先生成的图结构建模通道特征的相关性，并采用图注意力卷积对特征进行融合。

在实验部分，本章将上述两种模块组合设置了四种不同的配置，并在 OTB100、GOT-10k 和 LaSOT 三个数据集上进行测试与对比。根据实验结果，本章设计的算法相较于基

线算法都有了较高的性能提升，且在没有进行空间维度的相关运算的情况下性能在一些数据集上超过了进行空间维度的相关运算的跟踪算法，从实验的角度证明了利用图结构建模通道特征相关性并用图神经网络进行特征融合的相关运算技术的有效性。本章采用的图注意力卷积无法在硬件上直接找到已有的算子进行使用，第六章将会针对该算子进行替代实现。

第五章 基于分流注意力多尺度特征融合的孪生网络跟踪算法

前文所述的相关运算方式在空间维度和通道维度都进行了改进，而多尺度特征的相关运算虽然从直觉上会引入更丰富的特征进行相关运算，提高跟踪器适应目标尺度变化的能力，但是目前缺乏有效的实验证明。本章提出了一种基于分流注意力多尺度特征融合的孪生网络跟踪算法，在基于 Transformer 的相关运算中添加多尺度特征的提取与融合，观察多尺度特征的相关运算是否能够为最先进的相关运算带来性能提升。

5.1 引言

Transformer 是一种网络结构，最先[18]在自然语言处理领域进行使用并取得了极佳的性能表现，于是越来越多的工作尝试将 Transformer 网络应用于各种领域来处理各种数据，如时序数据[61]、图数据[62]、语音数据[63]和图像数据[23]等。在计算机视觉领域，最近的研究[23,64-70]表明视觉 Transformer 对于图像数据的学习和表征是具有重要意义的。

表 11 列举了最近使用视觉 Transformer 网络设计的特征提取网络在 ImageNet 数据集上完成图像分类任务的相关性能，其分类准确率指标超过了大部分深度卷积神经网络和多层感知机网络。

表 11 近年视觉 Transformer 网络在 ImageNet 分类任务上的表现

模型	参数量	FLOPs	Top-1 acc. (%)
DeiT-S[64]	22M	4.6G	79.8
DeiT-B[64]	86M	4.5G	81.8
PVT-Small[65]	25M	3.8G	79.8
PVT-Medium[65]	44M	6.7G	81.2
PVT-Large[65]	61M	9.8G	81.7
T2T-ViT-14[66]	22M	5.2G	81.5
T2T-ViT-19[66]	39M	8.9G	81.9
T2T-ViT-24[66]	64M	14.1G	82.3
TNT-S[67]	24M	5.2G	81.5
TNT-B[67]	66M	14.1G	82.9
iRPE-K[68]	87M	17.7G	82.4
iRPE-QKV[68]	22M	4.9G	81.4
GLiT-Small[69]	25M	4.4G	80.5

GLiT-Base[69]	96M	17.0G	82.3
Swin-T[70]	29M	4.5G	81.3
Swin-S[70]	50M	8.7G	83.0
Swin-B[70]	88M	15.4G	83.5
Shunted-T[23]	11.5M	2.1G	79.8
Shunted-S[23]	22.4M	4.9G	82.9
Shunted-B[23]	39.6M	8.1G	84.0

类似于卷积神经网络的研究思路，视觉 Transformer 网络结构的设计也越来越关注多尺度特征的提取与融合。如图 46 所示，展示了不同的工作提取和计算注意力机制中的查询算子（Query）、键算子（Key）和值算子（Value）的不同思路。

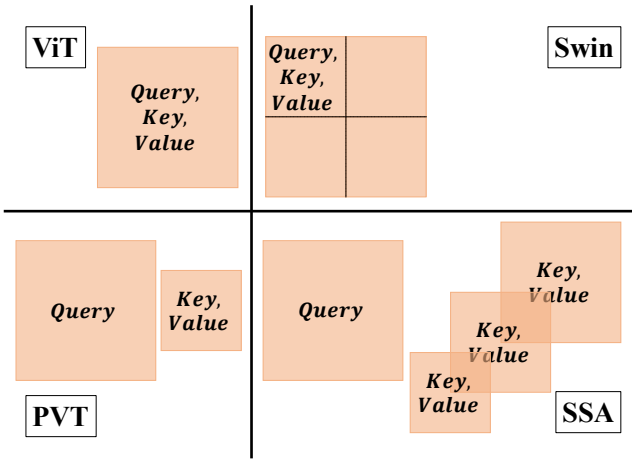

图 46　Transformer 中多尺度特征的研究

①ViT[19]：

早先的工作 ViT 在图像 token 上产生单一尺度的查询算子、键算子和值算子：

$$\begin{cases} \boldsymbol{q} = \mathrm{Linear}_q^{scale}(\boldsymbol{x}) \\ \boldsymbol{k} = \mathrm{Linear}_k^{scale}(\boldsymbol{x}) \\ \boldsymbol{v} = \mathrm{Linear}_v^{scale}(\boldsymbol{x}) \end{cases} \tag{5.1}$$

其中$\boldsymbol{x}$是图像的特征张量，Linear是线性变换操作，右上标一致的 scale 表示产生的尺度是一致的，右下标表示线性变换操作针对的算子。

采用单一尺度的自注意力机制将其进行融合：

$$\mathrm{Self\_Attention}(\boldsymbol{q},\boldsymbol{k},\boldsymbol{v}) = \mathrm{softmax}\left(\frac{\boldsymbol{q}\boldsymbol{k}^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{v} \tag{5.2}$$

②Swin Transformer[70]：

Swin Transformer 是近年非常成功的视觉 Transformer 网络之一，它将图像的特征张

量进行窗口划分，并进行单一尺度的查询算子、键算子和值算子的注意力机制计算。然而该工作设计了精妙的滑窗操作，使得不同窗口之间的查询算子、键算子和值算子可以进行交互，间接实现了不同尺度特征的注意力机制计算。

③PVT[65]：

PVT 在图像 token 上产生单一尺度的查询算子，以及单一但是不同尺度的键算子和值算子，同样采用自注意力机制将其进行融合：

$$\begin{cases} \boldsymbol{q} = \mathrm{Linear}_{q}^{scale1}(\boldsymbol{x}) \\ \boldsymbol{k} = \mathrm{Linear}_{k}^{scale2}(\boldsymbol{x}) \\ \boldsymbol{v} = \mathrm{Linear}_{v}^{scale2}(\boldsymbol{x}) \end{cases} \tag{5.3}$$

④SSA[23]：

SSA 在图像 token 上产生单一尺度的查询算子，以及多个不同尺度的键算子和值算子，同样采用自注意力机制将其进行融合：

$$\begin{cases} \boldsymbol{q} = \mathrm{Linear}_{q}^{scale1}(\boldsymbol{x}) \\ \boldsymbol{k_1} = \mathrm{Linear}_{k}^{scale1}(\boldsymbol{x}) \\ \boldsymbol{v_1} = \mathrm{Linear}_{v}^{scale1}(\boldsymbol{x}) \\ \boldsymbol{k_2} = \mathrm{Linear}_{k}^{scale2}(\boldsymbol{x}) \\ \boldsymbol{v_2} = \mathrm{Linear}_{v}^{scale2}(\boldsymbol{x}) \end{cases} \tag{5.4}$$

该方法被称为分流自注意力（Shunted Self-Attention, SSA）。

综上所述，学习图像数据的多尺度特征，有助于提升 Transformer 网络的性能表现，因此可以尝试将多尺度特征的学习融入基于 Transformer 网络的相关运算模块中，以提升跟踪算法适应跟踪目标尺度变化的能力。

本章目标是基于 Transformer 设计出一个能够有效融合模板图像特征和搜索图像特征两种图像特征张量的相关运算。由于 Transformer 网络的复杂性和特殊性，本章设计的相关运算不需要第三章和第四章中所采用的预融合模块，而可以直接处理模板图像特征 tokens 和搜索图像特征 tokens 并将其进行融合。本章设计的基于 Transformer 的相关运算模块，先采用分流自注意力机制设计分流自主语境增强（Shunted Ego-Context Augment, S-ECA）模块，再采用分流互注意力机制设计分流交叉特征增强（Shunted Cross-Feature Augment, S-CFA）模块，最后基于 S-ECA 模块和 S-CFA 模块对多尺度特征进行自适应地学习与融合。最后，设计的基于分流注意力多尺度特征融合的孪生网络跟踪算法将在 GOT-10k、YT-BB、COCO、DET 和 VID 五个数据集的训练集上进行训练，在

OTB100、GOT-10k 和 LaSOT 三个数据集的测试集上进行测试。

5.2 S-ECA 模块和 S-CFA 模块

本节将分流注意力机制引入文献[20]设计的自主语境增强模块与交叉特征增强模块，设计了分流自主语境增强模块 S-ECA 和分流交叉特征增强模块 S-CFA。这两个模块的输入都是来自图像特征张量拆解的 tokens 变换得到的查询算子、键算子和值算子，但是由于分流作用，这些算子分布包含了多尺度特征。

不妨设有模板图像特征的张量$\bar{\boldsymbol{z}}$和搜索图像特征的张量$\bar{\boldsymbol{x}}$。其中$\bar{\boldsymbol{z}} \in \mathbb{R}^{b\times c\times H_z\times W_z}$且$\bar{\boldsymbol{x}} \in \mathbb{R}^{b\times c\times H_x\times W_x}$。对$\bar{\boldsymbol{z}}$和$\bar{\boldsymbol{x}}$使用$1\times 1$卷积进行降维，得到$\bar{\boldsymbol{z}}_0$和$\bar{\boldsymbol{x}}_0$。其中$\bar{\boldsymbol{z}}_0 \in \mathbb{R}^{b\times d\times H_z\times W_z}$，$\bar{\boldsymbol{x}}_0 \in \mathbb{R}^{b\times d\times H_x\times W_x}$。S-ECA 模块的输入和 S-CFA 模块的输入即为$\bar{\boldsymbol{z}}_0$和$\bar{\boldsymbol{x}}_0$。

5.2.1 S-ECA 模块

分流自主语境增强模块 S-ECA 的主要功能是学习模板图像特征的多尺度增强表示，以及搜索图像特征的多尺度增强表示。如图 47 所示，本节设计的 S-ECA 模块使用的多头注意力机制的头数固定为 2，分 2 个支流进行多尺度特征增强，具体步骤分为下述六步：

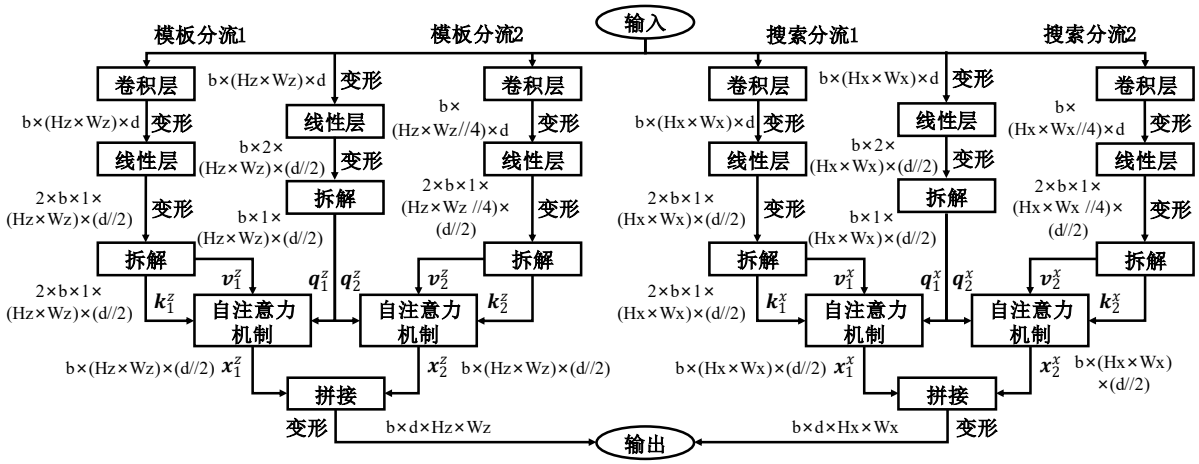

图 47　S-ECA 模块结构图

(1)模板分流的查询算子计算：

分流注意力机制对特征尺度进行分流，但共用同一尺度的查询算子。对于降维后的模板图像特征$\bar{\boldsymbol{z}}_0 \in \mathbb{R}^{b\times d\times H_z\times W_z}$，首先对它进行变形，得到$\bar{\boldsymbol{z}}_0 \in \mathbb{R}^{b\times (H_z\times W_z)\times d}$，再对$d$维的特征使用线性层进行特征变换：

$$\boldsymbol{q}^z = \mathrm{Linear}_q(\bar{\boldsymbol{z}}_0) \tag{5.5}$$

这里设置线性层Linear_q的输出特征维度同输入特征维度一致，均为d。得到的模板图像特征的查询算子$\boldsymbol{q}^z \in \mathbb{R}^{b\times(H_z\times W_z)\times d}$。考虑到头数为 2，将$\boldsymbol{q}^z$进行变形，得到：

$$\boldsymbol{q}^z \in \mathbb{R}^{b\times 2\times(H_z\times W_z)\times(d//2)}$$

其中$d//2$表示维度d除以 2 并向下取整。

将上述模板图像特征的查询算子$\boldsymbol{q}^z$按第二维进行拆分，以供 2 个分流分别进行自注意力计算：

$$\boldsymbol{q}_1^z, \boldsymbol{q}_2^z = \mathrm{Split}(\boldsymbol{q}^z) \tag{5.6}$$

其中，Split是拆分操作，$\boldsymbol{q}_1^z, \boldsymbol{q}_2^z \in \mathbb{R}^{b\times 1\times(H_z\times W_z)\times(d//2)}$，$\boldsymbol{q}_1^z$是第一支分流的模板特征查询算子，$\boldsymbol{q}_2^z$是第二个支流的模板特征查询算子。

(2)模板分流的键值算子计算：

分流注意力机制对特征尺度进行分流，使用不同尺度的键算子和值算子。对于降维后的模板图像特征$\bar{\boldsymbol{z}}_0 \in \mathbb{R}^{b\times d\times H_z\times W_z}$进行分流处理：

①第一支分流：

先使用一个卷积层对$\bar{\boldsymbol{z}}_0$进行尺度特征提取：

$$\bar{\boldsymbol{z}}_0^1 = \mathrm{Conv2d}_1(\bar{\boldsymbol{z}}_0) \tag{5.7}$$

其中，$\mathrm{Conv2d}_1$是第一支分流的尺度特征二维卷积层，$\bar{\boldsymbol{z}}_0^1 \in \mathbb{R}^{b\times d\times H_z\times W_z}$，即该尺度和模板特征的查询算子尺度一致。

为了便于进行线性变换，对$\bar{\boldsymbol{z}}_0^1$进行变形操作，得到$\bar{\boldsymbol{z}}_0^1 \in \mathbb{R}^{b\times(H_z\times W_z)\times d}$，再对$d$维的特征使用线性层进行特征变换：

$$(\boldsymbol{kv})_1^z = \mathrm{Linear}_{kv}(\bar{\boldsymbol{z}}_0^1) \tag{5.8}$$

这里设置线性层Linear_{kv}的输出特征维度同输入特征维度一致，均为d。得到的第一支分流的模板图像特征的键值算子$(\boldsymbol{kv})_1^z \in \mathbb{R}^{b\times(H_z\times W_z)\times d}$。

考虑到头数为 2，将$(\boldsymbol{kv})_1^z$进行变形，得到：

$$(\boldsymbol{kv})_1^z \in \mathbb{R}^{2\times b\times 1\times(H_z\times W_z)\times(d//2)}$$

将上述第一支分流的模板图像特征的键值算子$(\boldsymbol{kv})_1^z$按第一维进行拆分，以得到第一支分流的键算子和值算子：

$$\boldsymbol{k}_1^z, \boldsymbol{v}_1^z = \mathrm{Split}((\boldsymbol{kv})_1^z) \tag{5.9}$$

其中，Split是拆分操作，$\boldsymbol{k}_1^z, \boldsymbol{v}_1^z \in \mathbb{R}^{b\times 1\times(H_z\times W_z)\times(d//2)}$，$\boldsymbol{k}_1^z$是第一支分流的模板特征键算

子，$\boldsymbol{v}_1^z$是第一支分流的模板特征值算子。

②第二支分流：

先使用一个卷积层对$\bar{\boldsymbol{z}}_0$进行尺度特征提取：

$$\bar{\boldsymbol{z}}_0^2 = \mathrm{Conv2d}_2(\bar{\boldsymbol{z}}_0) \tag{5.10}$$

其中，$\mathrm{Conv2d}_2$是第二支分流的尺度特征二维卷积层，$\bar{\boldsymbol{z}}_0^2 \in \mathbb{R}^{b\times d\times(H_z//2)\times(W_z//2)}$，即该尺度相较于第一支分流的特征长宽都减半。

为了便于进行线性变换，对$\bar{\boldsymbol{z}}_0^2$进行变形操作，得到$\bar{\boldsymbol{z}}_0^2 \in \mathbb{R}^{b\times(H_z\times W_z//4)\times d}$，再对$\boldsymbol{d}$维的特征使用线性层进行特征变换：

$$(\boldsymbol{kv})_2^z = \mathrm{Linear}_{kv}(\bar{\boldsymbol{z}}_0^2) \tag{5.11}$$

这里设置线性层Linear_{kv}的输出特征维度同输入特征维度一致，均为$\boldsymbol{d}$。得到的第二支分流的模板图像特征的键值算子$(\boldsymbol{kv})_2^z \in \mathbb{R}^{b\times(H_z\times W_z//4)\times d}$。

考虑到头数为 2，将$(\boldsymbol{kv})_2^z$进行变形，得到：

$$(\boldsymbol{kv})_2^z \in \mathbb{R}^{2\times b\times 1\times(H_z\times W_z//4)\times(d//2)}$$

将上述第二支分流的模板图像特征的键值算子$(\boldsymbol{kv})_2^z$按第一维进行拆分，以得到第二支分流的键算子和值算子：

$$\boldsymbol{k}_2^z, \boldsymbol{v}_2^z = \mathrm{Split}((\boldsymbol{kv})_2^z) \tag{5.12}$$

其中，Split是拆分操作，$\boldsymbol{k}_2^z, \boldsymbol{v}_2^z \in \mathbb{R}^{b\times 1\times(H_z\times W_z//4)\times(d//2)}$，$\boldsymbol{k}_2^z$是第二支分流的模板特征键算子，$\boldsymbol{v}_2^z$是第二支分流的模板特征值算子。

(3)模板分流的多头自注意力机制计算：

对两支分流的查询算子、键算子和值算子分别进行多头自注意力机制计算：

①第一支分流：

$$\boldsymbol{x}_1^z = \mathrm{Self\_Attention}(\boldsymbol{q}_1^z, \boldsymbol{k}_1^z, \boldsymbol{v}_1^z) = \mathrm{Softmax}\left(\frac{\boldsymbol{q}_1^z {\boldsymbol{k}_1^z}^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{v}_1^z \tag{5.13}$$

其中，$\boldsymbol{q}_1^z {\boldsymbol{k}_1^z}^{\mathrm{T}} \in \mathbb{R}^{b\times 1\times(H_z\times W_z)\times(H_z\times W_z)}$，Softmax是 Softmax 函数，$d_k$是缩放因子，一般设置为$\boldsymbol{k}_1^z$的维度。

第一支分流的自注意力机制计算结果为：

$$\boldsymbol{x}_1^z \in \mathbb{R}^{b\times 1\times(H_z\times W_z)\times(d//2)}$$

②第二支分流：

$$\boldsymbol{x}_2^z = \text{Self\_Attention}(\boldsymbol{q}_2^z, \boldsymbol{k}_2^z, \boldsymbol{v}_2^z) = \text{softmax}\left(\frac{\boldsymbol{q}_2^z \boldsymbol{k}_2^{z\text{T}}}{\sqrt{d_k}}\right)\boldsymbol{v}_2^z \tag{5.14}$$

其中，$\boldsymbol{q}_2^z \boldsymbol{k}_2^{z\text{T}} \in \mathbb{R}^{b\times 1\times(H_z\times W_z)\times(H_z\times W_z//4)}$。

第二支分流的计算结果为：

$$\boldsymbol{x}_2^z \in \mathbb{R}^{b\times 1\times(H_z\times W_z)\times(d//2)}$$

③分流汇合：

将$\boldsymbol{x}_1^z$和$\boldsymbol{x}_2^z$在特征维度进行拼接：

$$\boldsymbol{x}^z = \text{Concat}(\boldsymbol{x}_1^z, \boldsymbol{x}_2^z) \tag{5.15}$$

其中，Concat是拼接操作，$\boldsymbol{x}^z \in \mathbb{R}^{b\times 1\times(H_z\times W_z)\times d}$。

最后将$\boldsymbol{x}^z$变形得到模板图像多尺度特征增强结果：

$$\boldsymbol{x}^z \in \mathbb{R}^{b\times d\times H_z\times W_z}$$

分流汇合完成了模板图像不同尺度的特征增强。

(4)搜索分流的查询算子计算：

分流注意力机制对特征尺度进行分流，但共用同一尺度的查询算子。对于降维后的搜索图像特征$\overline{\boldsymbol{x}}_0 \in \mathbb{R}^{b\times d\times H_x\times W_x}$，首先对它进行变形，得到$\overline{\boldsymbol{x}}_0 \in \mathbb{R}^{b\times(H_x\times W_x)\times d}$，再对$d$维的特征使用线性层进行特征变换：

$$\boldsymbol{q}^x = \text{Linear}_q(\overline{\boldsymbol{x}}_0) \tag{5.16}$$

这里设置线性层Linear_q的输出特征维度同输入特征维度一致，均为d。得到的搜索图像特征的查询算子$\boldsymbol{q}^x \in \mathbb{R}^{b\times(H_x\times W_x)\times d}$。考虑到头数为 2，将$\boldsymbol{q}^x$进行变形，得到：

$$\boldsymbol{q}^x \in \mathbb{R}^{b\times 2\times(H_x\times W_x)\times(d//2)}$$

其中$d//2$表示维度d除以 2 并向下取整。

将上述搜索图像特征的查询算子$\boldsymbol{q}^x$按第二维进行拆分，以供 2 个分流分别进行自注意力计算：

$$\boldsymbol{q}_1^x, \boldsymbol{q}_2^x = \text{Split}(\boldsymbol{q}^x) \tag{5.17}$$

其中，Split是拆分操作，$\boldsymbol{q}_1^x, \boldsymbol{q}_2^x \in \mathbb{R}^{b\times 1\times(H_x\times W_x)\times(d//2)}$，$\boldsymbol{q}_1^x$是第一支分流的搜索特征查询算子，$\boldsymbol{q}_2^x$是第二个支流的搜索特征查询算子。

(5)搜索分流的键值算子计算：

分流注意力机制对特征尺度进行分流，使用不同尺度的键算子和值算子。对于降维

后的搜索图像特征$\overline{\boldsymbol{x}}_0 \in \mathbb{R}^{b\times d\times H_x\times W_x}$进行分流处理：

①第一支分流：

先使用一个卷积层对$\overline{\boldsymbol{x}}_0$进行尺度特征提取：

$$\overline{\boldsymbol{x}}_0^1 = \mathrm{Conv2d}_1(\overline{\boldsymbol{x}}_0) \tag{5.18}$$

其中，$\mathrm{Conv2d}_1$是第一支分流的尺度特征二维卷积层，$\overline{\boldsymbol{x}}_0^1 \in \mathbb{R}^{b\times d\times H_x\times W_x}$，即该尺度和搜索特征的查询算子尺度一致。

为了便于进行线性变换，对$\overline{\boldsymbol{x}}_0^1$进行变形操作，得到$\overline{\boldsymbol{x}}_0^1 \in \mathbb{R}^{b\times (H_x\times W_x)\times d}$，再对$\boldsymbol{d}$维的特征使用线性层进行特征变换：

$$(\boldsymbol{kv})_1^x = \mathrm{Linear}_{kv}(\overline{\boldsymbol{x}}_0^1) \tag{5.19}$$

这里设置线性层Linear_{kv}的输出特征维度同输入特征维度一致，均为$\boldsymbol{d}$。得到的第一支分流的搜索图像特征的键值算子$(\boldsymbol{kv})_1^x \in \mathbb{R}^{b\times (H_x\times W_x)\times d}$。

考虑到头数为 2，将$(\boldsymbol{kv})_1^x$进行变形，得到：

$$(\boldsymbol{kv})_1^x \in \mathbb{R}^{2\times b\times 1\times (H_x\times W_x)\times (d//2)}$$

将上述第一支分流的搜索图像特征的键值算子$(\boldsymbol{kv})_1^x$按第一维进行拆分，以得到第一支分流的键算子和值算子：

$$\boldsymbol{k}_1^x, \boldsymbol{v}_1^x = \mathrm{Split}((\boldsymbol{kv})_1^x) \tag{5.20}$$

其中，Split是拆分操作，$\boldsymbol{k}_1^x, \boldsymbol{v}_1^x \in \mathbb{R}^{b\times 1\times (H_x\times W_x)\times (d//2)}$，$\boldsymbol{k}_1^x$是第一支分流的搜索特征键算子，$\boldsymbol{v}_1^x$是第一支分流的搜索特征值算子。

②第二支分流：

先使用一个卷积层对$\overline{\boldsymbol{x}}_0$进行尺度特征提取：

$$\overline{\boldsymbol{x}}_0^2 = \mathrm{Conv2d}_2(\overline{\boldsymbol{x}}_0) \tag{5.21}$$

其中，$\mathrm{Conv2d}_2$是第二支分流的尺度特征二维卷积层，$\overline{\boldsymbol{x}}_0^2 \in \mathbb{R}^{b\times d\times (H_x//2)\times (W_x//2)}$，即该尺度相较于第一支分流的特征长宽都减半。

为了便于进行线性变换，对$\overline{\boldsymbol{x}}_0^2$进行变形操作，得到$\overline{\boldsymbol{x}}_0^2 \in \mathbb{R}^{b\times (H_x\times W_x//4)\times d}$，再对$\boldsymbol{d}$维的特征使用线性层进行特征变换：

$$(\boldsymbol{kv})_2^x = \mathrm{Linear}_{kv}(\overline{\boldsymbol{x}}_0^2) \tag{5.22}$$

这里设置线性层Linear_{kv}的输出特征维度同输入特征维度一致，均为$\boldsymbol{d}$。得到的第二支分流的搜索图像特征的键值算子$(\boldsymbol{kv})_2^x \in \mathbb{R}^{b\times (H_x\times W_x//4)\times d}$。

考虑到头数为2，将$(\boldsymbol{kv})_2^x$进行变形，得到：

$$(\boldsymbol{kv})_2^x \in \mathbb{R}^{2\times b\times 1\times (H_x\times W_x//4)\times (d//2)}$$

将上述第二支分流的搜索图像特征的键值算子$(\boldsymbol{kv})_2^x$按第一维进行拆分，以得到第二支分流的键算子和值算子：

$$\boldsymbol{k}_2^x, \boldsymbol{v}_2^x = \text{Split}((\boldsymbol{kv})_2^x) \tag{5.23}$$

其中，Split是拆分操作，$\boldsymbol{k}_2^x, \boldsymbol{v}_2^x \in \mathbb{R}^{b\times 1\times (H_x\times W_x//4)\times (d//2)}$，$\boldsymbol{k}_2^x$是第二支分流的搜索特征键算子，$\boldsymbol{v}_2^x$是第二支分流的搜索特征值算子。

(6)搜索分流的多头自注意力机制计算：

对两支分流的查询算子、键算子和值算子分别进行多头自注意力机制计算：

①第一支分流：

$$\boldsymbol{x}_1^x = \text{Self\_Attention}(\boldsymbol{q}_1^x, \boldsymbol{k}_1^x, \boldsymbol{v}_1^x) = \text{Softmax}\left(\frac{\boldsymbol{q}_1^x \boldsymbol{k}_1^{x\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{v}_1^x \tag{5.24}$$

其中，$\boldsymbol{q}_1^x \boldsymbol{k}_1^{x\mathrm{T}} \in \mathbb{R}^{b\times 1\times (H_x\times W_x)\times (H_x\times W_x)}$，Softmax是 Softmax 函数，$d_k$是缩放因子，一般设置为$\boldsymbol{k}_1^x$的维度。

第一支分流的自注意力机制计算结果为：

$$\boldsymbol{x}_1^x \in \mathbb{R}^{b\times 1\times (H_x\times W_x)\times (d//2)}$$

②第二支分流：

$$\boldsymbol{x}_2^x = \text{Self\_Attention}(\boldsymbol{q}_2^x, \boldsymbol{k}_2^x, \boldsymbol{v}_2^x) = \text{softmax}\left(\frac{\boldsymbol{q}_2^x \boldsymbol{k}_2^{x\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{v}_2^x \tag{5.25}$$

其中，$\boldsymbol{q}_2^x \boldsymbol{k}_2^{x\mathrm{T}} \in \mathbb{R}^{b\times 1\times (H_x\times W_x)\times (H_x\times W_x//4)}$。

第二支分流的计算结果为：

$$\boldsymbol{x}_2^x \in \mathbb{R}^{b\times 1\times (H_x\times W_x)\times (d//2)}$$

③分流汇合：

将$\boldsymbol{x}_1^x$和$\boldsymbol{x}_2^x$在特征维度进行拼接：

$$\boldsymbol{x}^x = \text{Concat}(\boldsymbol{x}_1^x, \boldsymbol{x}_2^x) \tag{5.26}$$

其中，Concat是拼接操作，$\boldsymbol{x}^x \in \mathbb{R}^{b\times 1\times (H_x\times W_x)\times d}$。

最后将$\boldsymbol{x}^x$变形得到搜索图像多尺度特征增强结果：

$$\boldsymbol{x}^x \in \mathbb{R}^{b\times d\times H_x\times W_x}$$

分流汇合完成了搜索图像不同尺度的特征增强。

综上六步完成 S-ECA 模块的计算。

5.2.2 S-CFA 模块

分流交叉特征增强模块的主要功能是学习模板图像特征和搜索图像特征的多尺度融合表示。如图 48 所示，本节设计的 S-CFA 模块使用的多头注意力机制的头数固定为 2，分 2 个支流进行多尺度特征融合，具体步骤分为下述六步：

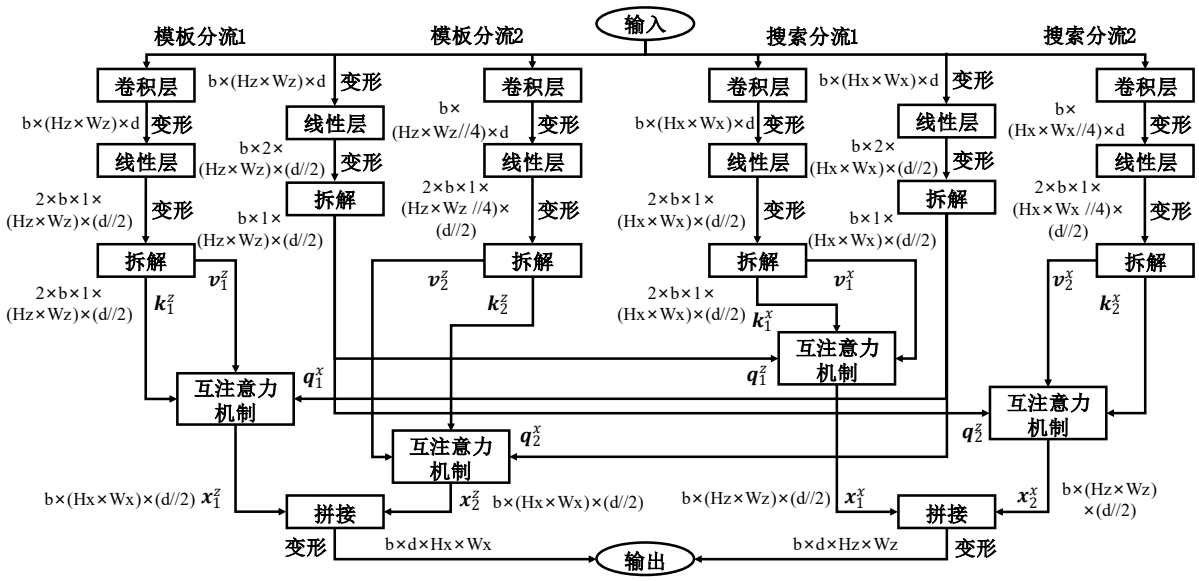

图 48　S-CFA 模块结构图

(1)模板分流的查询算子计算：

分流注意力机制对特征尺度进行分流，但共用同一尺度的查询算子。对于降维后的模板图像特征$\bar{\boldsymbol{z}}_0 \in \mathbb{R}^{b\times d\times H_z\times W_z}$，首先对它进行变形，得到$\bar{\boldsymbol{z}}_0 \in \mathbb{R}^{b\times (H_z\times W_z)\times d}$，再对$d$维的特征使用线性层进行特征变换：

$$\boldsymbol{q}^z = \mathrm{Linear}_q(\bar{\boldsymbol{z}}_0) \tag{5.27}$$

这里设置线性层Linear_q的输出特征维度同输入特征维度一致，均为d。得到的模板图像特征的查询算子$\boldsymbol{q}^z \in \mathbb{R}^{b\times (H_z\times W_z)\times d}$。考虑到头数为 2，将$\boldsymbol{q}^z$进行变形，得到：

$$\boldsymbol{q}^z \in \mathbb{R}^{b\times 2\times (H_z\times W_z)\times (d//2)}$$

其中$d//2$表示维度d除以 2 并向下取整。

将上述模板图像特征的查询算子$\boldsymbol{q}^z$按第二维进行拆分，以供 2 个分流分别进行自注意力计算：

$$\boldsymbol{q}_1^z, \boldsymbol{q}_2^z = \mathrm{Split}(\boldsymbol{q}^z) \tag{5.28}$$

其中，Split是拆分操作，$\boldsymbol{q}_1^z, \boldsymbol{q}_2^z \in \mathbb{R}^{b\times 1\times (H_z\times W_z)\times (d//2)}$，$\boldsymbol{q}_1^z$是第一支分流的模板特征查询算子，$\boldsymbol{q}_2^z$是第二个支流的模板特征查询算子。

(2)模板分流的键值算子计算：

分流注意力机制对特征尺度进行分流，使用不同尺度的键算子和值算子。对于降维后的模板图像特征$\bar{\boldsymbol{z}}_0 \in \mathbb{R}^{b\times d\times H_z\times W_z}$进行分流处理：

①第一支分流：

先使用一个卷积层对$\bar{\boldsymbol{z}}_0$进行尺度特征提取：

$$\bar{\boldsymbol{z}}_0^1 = \mathrm{Conv2d}_1(\bar{\boldsymbol{z}}_0) \tag{5.29}$$

其中，$\mathrm{Conv2d}_1$是第一支分流的尺度特征二维卷积层，$\bar{\boldsymbol{z}}_0^1 \in \mathbb{R}^{b\times d\times H_z\times W_z}$，即该尺度和模板特征的查询算子尺度一致。

为了便于进行线性变换，对$\bar{\boldsymbol{z}}_0^1$进行变形操作，得到$\bar{\boldsymbol{z}}_0^1 \in \mathbb{R}^{b\times (H_z\times W_z)\times d}$，再对$d$维的特征使用线性层进行特征变换：

$$(\boldsymbol{kv})_1^z = \mathrm{Linear}_{kv}(\bar{\boldsymbol{z}}_0^1) \tag{5.30}$$

这里设置线性层Linear_{kv}的输出特征维度同输入特征维度一致，均为d。得到的第一支分流的模板图像特征的键值算子$(\boldsymbol{kv})_1^z \in \mathbb{R}^{b\times (H_z\times W_z)\times d}$。

考虑到头数为 2，将$(\boldsymbol{kv})_1^z$进行变形，得到：

$$(\boldsymbol{kv})_1^z \in \mathbb{R}^{2\times b\times 1\times (H_z\times W_z)\times (d//2)}$$

将上述第一支分流的模板图像特征的键值算子$(\boldsymbol{kv})_1^z$按第一维进行拆分，以得到第一支分流的键算子和值算子：

$$\boldsymbol{k}_1^z, \boldsymbol{v}_1^z = \mathrm{Split}((\boldsymbol{kv})_1^z) \tag{5.31}$$

其中，Split是拆分操作，$\boldsymbol{k}_1^z, \boldsymbol{v}_1^z \in \mathbb{R}^{b\times 1\times (H_z\times W_z)\times (d//2)}$，$\boldsymbol{k}_1^z$是第一支分流的模板特征键算子，$\boldsymbol{v}_1^z$是第一支分流的模板特征值算子。

②第二支分流：

先使用一个卷积层对$\bar{\boldsymbol{z}}_0$进行尺度特征提取：

$$\bar{\boldsymbol{z}}_0^2 = \mathrm{Conv2d}_2(\bar{\boldsymbol{z}}_0) \tag{5.32}$$

其中，$\mathrm{Conv2d}_2$是第二支分流的尺度特征二维卷积层，$\bar{\boldsymbol{z}}_0^2 \in \mathbb{R}^{b\times d\times (H_z//2)\times (W_z//2)}$，即该尺度相较于第一支分流的特征长宽都减半。

为了便于进行线性变换，对$\bar{\boldsymbol{z}}_0^2$进行变形操作，得到$\bar{\boldsymbol{z}}_0^2 \in \mathbb{R}^{b\times (H_z\times W_z//4)\times d}$，再对$d$维的

特征使用线性层进行特征变换：

$$(\boldsymbol{kv})_2^z = \text{Linear}_{kv}(\overline{\boldsymbol{z}}_0^2) \tag{5.33}$$

这里设置线性层Linear_{kv}的输出特征维度同输入特征维度一致，均为d。得到的第二支分流的模板图像特征的键值算子$(\boldsymbol{kv})_2^z \in \mathbb{R}^{b\times(H_z\times W_z//4)\times d}$。

考虑到头数为 2，将$(\boldsymbol{kv})_2^z$进行变形，得到：

$$(\boldsymbol{kv})_2^z \in \mathbb{R}^{2\times b\times 1\times(H_z\times W_z//4)\times(d//2)}$$

将上述第二支分流的模板图像特征的键值算子$(\boldsymbol{kv})_2^z$按第一维进行拆分，以得到第二支分流的键算子和值算子：

$$\boldsymbol{k}_2^z, \boldsymbol{v}_2^z = \text{Split}((\boldsymbol{kv})_2^z) \tag{5.34}$$

其中，Split是拆分操作，$\boldsymbol{k}_2^z, \boldsymbol{v}_2^z \in \mathbb{R}^{b\times 1\times(H_z\times W_z//4)\times(d//2)}$，$\boldsymbol{k}_2^z$是第二支分流的模板特征键算子，$\boldsymbol{v}_2^z$是第二支分流的模板特征值算子。

(3)搜索分流的查询算子计算：

分流注意力机制对特征尺度进行分流，但共用同一尺度的查询算子。对于降维后的搜索图像特征$\overline{\boldsymbol{x}}_0 \in \mathbb{R}^{b\times d\times H_x\times W_x}$，首先对它进行变形，得到$\overline{\boldsymbol{x}}_0 \in \mathbb{R}^{b\times(H_x\times W_x)\times d}$，再对$d$维的特征使用线性层进行特征变换：

$$\boldsymbol{q}^x = \text{Linear}_q(\overline{\boldsymbol{x}}_0) \tag{5.35}$$

这里设置线性层Linear_q的输出特征维度同输入特征维度一致，均为d。得到的搜索图像特征的查询算子$\boldsymbol{q}^x \in \mathbb{R}^{b\times(H_x\times W_x)\times d}$。考虑到头数为 2，将$\boldsymbol{q}^x$进行变形，得到：

$$\boldsymbol{q}^x \in \mathbb{R}^{b\times 2\times(H_x\times W_x)\times(d//2)}$$

其中$d//2$表示维度d除以 2 并向下取整。

将上述搜索图像特征的查询算子$\boldsymbol{q}^x$按第二维进行拆分，以供 2 个分流分别进行自注意力计算：

$$\boldsymbol{q}_1^x, \boldsymbol{q}_2^x = \text{Split}(\boldsymbol{q}^x) \tag{5.36}$$

其中，Split是拆分操作，$\boldsymbol{q}_1^x, \boldsymbol{q}_2^x \in \mathbb{R}^{b\times 1\times(H_x\times W_x)\times(d//2)}$，$\boldsymbol{q}_1^x$是第一支分流的搜索特征查询算子，$\boldsymbol{q}_2^x$是第二个支流的搜索特征查询算子。

(4)搜索分流的键值算子计算：

分流注意力机制对特征尺度进行分流，使用不同尺度的键算子和值算子。对于降维后的搜索图像特征$\overline{\boldsymbol{x}}_0 \in \mathbb{R}^{b\times d\times H_x\times W_x}$进行分流处理：

①第一支分流：

先使用一个卷积层对$\overline{\boldsymbol{x}}_0$进行尺度特征提取：

$$\overline{\boldsymbol{x}}_0^1 = \mathrm{Conv2d}_1(\overline{\boldsymbol{x}}_0) \tag{5.37}$$

其中，$\mathrm{Conv2d}_1$是第一支分流的尺度特征二维卷积层，$\overline{\boldsymbol{x}}_0^1 \in \mathbb{R}^{b\times d\times H_x\times W_x}$，即该尺度和搜索特征的查询算子尺度一致。

为了便于进行线性变换，对$\overline{\boldsymbol{x}}_0^1$进行变形操作，得到$\overline{\boldsymbol{x}}_0^1 \in \mathbb{R}^{b\times (H_x\times W_x)\times d}$，再对$d$维的特征使用线性层进行特征变换：

$$(\boldsymbol{kv})_1^x = \mathrm{Linear}_{kv}(\overline{\boldsymbol{x}}_0^1) \tag{5.38}$$

这里设置线性层Linear_{kv}的输出特征维度同输入特征维度一致，均为d。得到的第一支分流的搜索图像特征的键值算子$(\boldsymbol{kv})_1^x \in \mathbb{R}^{b\times (H_x\times W_x)\times d}$。

考虑到头数为 2，将$(\boldsymbol{kv})_1^x$进行变形，得到：

$$(\boldsymbol{kv})_1^x \in \mathbb{R}^{2\times b\times 1\times (H_x\times W_x)\times (d//2)}$$

将上述第一支分流的搜索图像特征的键值算子$(\boldsymbol{kv})_1^x$按第一维进行拆分，以得到第一支分流的键算子和值算子：

$$\boldsymbol{k}_1^x, \boldsymbol{v}_1^x = \mathrm{Split}((\boldsymbol{kv})_1^x) \tag{5.39}$$

其中，Split是拆分操作，$\boldsymbol{k}_1^x, \boldsymbol{v}_1^x \in \mathbb{R}^{b\times 1\times (H_x\times W_x)\times (d//2)}$，$\boldsymbol{k}_1^x$是第一支分流的搜索特征键算子，$\boldsymbol{v}_1^x$是第一支分流的搜索特征值算子。

②第二支分流：

先使用一个卷积层对$\overline{\boldsymbol{x}}_0$进行尺度特征提取：

$$\overline{\boldsymbol{x}}_0^2 = \mathrm{Conv2d}_2(\overline{\boldsymbol{x}}_0) \tag{5.40}$$

其中，$\mathrm{Conv2d}_2$是第二支分流的尺度特征二维卷积层，$\overline{\boldsymbol{x}}_0^2 \in \mathbb{R}^{b\times d\times (H_x//2)\times (W_x//2)}$，即该尺度相较于第一支分流的特征长宽都减半。

为了便于进行线性变换，对$\overline{\boldsymbol{x}}_0^2$进行变形操作，得到$\overline{\boldsymbol{x}}_0^2 \in \mathbb{R}^{b\times (H_x\times W_x//4)\times d}$，再对$d$维的特征使用线性层进行特征变换：

$$(\boldsymbol{kv})_2^x = \mathrm{Linear}_{kv}(\overline{\boldsymbol{x}}_0^2) \tag{5.41}$$

这里设置线性层Linear_{kv}的输出特征维度同输入特征维度一致，均为d。得到的第二支分流的搜索图像特征的键值算子$(\boldsymbol{kv})_2^x \in \mathbb{R}^{b\times (H_x\times W_x//4)\times d}$。

考虑到头数为 2，将$(\boldsymbol{kv})_2^x$进行变形，得到：

$$(\boldsymbol{kv})_2^x \in \mathbb{R}^{2\times b\times 1\times (H_x\times W_x//4)\times (d//2)}$$

将上述第二支分流的搜索图像特征的键值算子$(\boldsymbol{kv})_2^x$按第一维进行拆分，以得到第二支分流的键算子和值算子：

$$\boldsymbol{k}_2^x, \boldsymbol{v}_2^x = \text{Split}((\boldsymbol{kv})_2^x) \tag{5.42}$$

其中，Split是拆分操作，$\boldsymbol{k}_2^x, \boldsymbol{v}_2^x \in \mathbb{R}^{b\times 1\times (H_x\times W_x//4)\times (d//2)}$，$\boldsymbol{k}_2^x$是第二支分流的搜索特征键算子，$\boldsymbol{v}_2^x$是第二支分流的搜索特征值算子。

(5)“模板-搜索”分流的多头互注意力机制计算：

对两支分流的模板特征查询算子、搜索特征键算子和搜索特征值算子分别进行多头互注意力机制计算：

①第一支分流：

$$\boldsymbol{x}_1^{zx} = \text{Cross\_Attention}(\boldsymbol{q}_1^z, \boldsymbol{k}_1^x, \boldsymbol{v}_1^x) = \text{Softmax}\left(\frac{\boldsymbol{q}_1^z \boldsymbol{k}_1^{x\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{v}_1^x \tag{5.43}$$

其中，$\boldsymbol{q}_1^z \boldsymbol{k}_1^{x\mathrm{T}} \in \mathbb{R}^{b\times 1\times (H_z\times W_z)\times (H_x\times W_x)}$，Softmax是 Softmax 函数，$d_k$是缩放因子，一般设置为$\boldsymbol{k}_1^x$的维度。

第一支分流的互注意力机制计算结果为：

$$\boldsymbol{x}_1^{zx} \in \mathbb{R}^{b\times 1\times (H_z\times W_z)\times (d//2)}$$

②第二支分流：

$$\boldsymbol{x}_2^{zx} = \text{Cross\_Attention}(\boldsymbol{q}_2^z, \boldsymbol{k}_2^x, \boldsymbol{v}_2^x) = \text{softmax}\left(\frac{\boldsymbol{q}_2^z \boldsymbol{k}_2^{x\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{v}_2^x \tag{5.44}$$

其中，$\boldsymbol{q}_2^z \boldsymbol{k}_2^{x\mathrm{T}} \in \mathbb{R}^{b\times 1\times (H_z\times W_z)\times (H_x\times W_x//4)}$。

第二支分流的计算结果为：

$$\boldsymbol{x}_2^{zx} \in \mathbb{R}^{b\times 1\times (H_z\times W_z)\times (d//2)}$$

③分流汇合：

将$\boldsymbol{x}_1^{zx}$和$\boldsymbol{x}_2^{zx}$在特征维度进行拼接：

$$\boldsymbol{x}^{zx} = \text{Concat}(\boldsymbol{x}_1^{zx}, \boldsymbol{x}_2^{zx}) \tag{5.45}$$

其中，Concat是拼接操作，$\boldsymbol{x}^{zx} \in \mathbb{R}^{b\times 1\times (H_z\times W_z)\times d}$。

最后将$\boldsymbol{x}^{zx}$变形得到模板图像特征和搜索图像特征进行多尺度特征融合结果：

$$\boldsymbol{x}^{zx} \in \mathbb{R}^{b\times d\times H_z\times W_z}$$

分流汇合完成了模板图像与搜索图像不同尺度的特征融合。

(6) “搜索-模板”分流的多头互注意力机制计算：

对两支分流的搜索特征查询算子、模板特征键算子和模板特征值算子分别进行多头互注意力机制计算：

①第一支分流：

$$\boldsymbol{x}_1^{xz} = \text{Cross\_Attention}(\boldsymbol{q}_1^x, \boldsymbol{k}_1^z, \boldsymbol{v}_1^z) = \text{Softmax}\left(\frac{\boldsymbol{q}_1^x \boldsymbol{k}_1^{z\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{v}_1^z \tag{5.46}$$

其中，$\boldsymbol{q}_1^x \boldsymbol{k}_1^{z\mathrm{T}} \in \mathbb{R}^{b\times 1\times(H_x\times W_x)\times(H_z\times W_z)}$，Softmax是 Softmax 函数，$d_k$是缩放因子，一般设置为$\boldsymbol{k}_1^z$的维度。

第一支分流的互注意力机制计算结果为：

$$\boldsymbol{x}_1^{xz} \in \mathbb{R}^{b\times 1\times(H_x\times W_x)\times(d//2)}$$

②第二支分流：

$$\boldsymbol{x}_2^{xz} = \text{Cross\_Attention}(\boldsymbol{q}_2^x, \boldsymbol{k}_2^z, \boldsymbol{v}_2^z) = \text{softmax}\left(\frac{\boldsymbol{q}_2^x \boldsymbol{k}_2^{z\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{v}_2^z \tag{5.47}$$

其中，$\boldsymbol{q}_2^x \boldsymbol{k}_2^{z\mathrm{T}} \in \mathbb{R}^{b\times 1\times(H_x\times W_x)\times(H_z\times W_z//4)}$。

第二支分流的计算结果为：

$$\boldsymbol{x}_2^{xz} \in \mathbb{R}^{b\times 1\times(H_x\times W_x)\times(d//2)}$$

③分流汇合：

将$\boldsymbol{x}_1^{xz}$和$\boldsymbol{x}_2^{xz}$在特征维度进行拼接：

$$\boldsymbol{x}^{xz} = \text{Concat}(\boldsymbol{x}_1^{xz}, \boldsymbol{x}_2^{xz}) \tag{5.48}$$

其中，Concat是拼接操作，$\boldsymbol{x}^{xz} \in \mathbb{R}^{b\times 1\times(H_x\times W_x)\times d}$。

最后将$\boldsymbol{x}^{xz}$得到搜索图像特征和模板图像特征进行多尺度特征融合结果：

$$\boldsymbol{x}^{xz} \in \mathbb{R}^{b\times d\times H_x\times W_x}$$

分流汇合完成了搜索图像与模板图像不同尺度的特征融合。

综上六步完成 S-CFA 模块的计算。

5.3 Transformer 相关运算模块

本节设计的 Transformer 相关运算模块的基本处理流程如图 49 所示，在该相关运算

模块中，使用不断堆叠的 S-ECA 模块和 S-CFA 模块，增强与融合来自模板图像特征的 tokens 和来自搜索图像特征的 tokens。

设该模块的输入为模板图像特征$\bar{\boldsymbol{z}} \in \mathbb{R}^{b\times c\times H_z\times W_z}$和搜索图像特征$\bar{\boldsymbol{x}} \in \mathbb{R}^{b\times c\times H_x\times W_x}$，具体过程通过重复下述 2 步完成相关运算，其中第一步利用 S-ECA 模块，第二步利用 S-CFA 模块：

(1)模板图像特征与搜索图像特征自增强：

将模板图像特征$\bar{\boldsymbol{z}}$和搜索图像特征$\bar{\boldsymbol{x}}$输入至一个 S-ECA 模块中，得到自增强的输出结果：

$$\boldsymbol{x}^z \in \mathbb{R}^{b\times c\times H_z\times W_z}, \boldsymbol{x}^x \in \mathbb{R}^{b\times c\times H_x\times W_x}$$

(2)模板图像特征与搜索图像特征互融合：

将上一步得到结果$\boldsymbol{x}^z$和$\boldsymbol{x}^x$输入至一个 S-CFA 模块中，得到多尺度特征互相融合的结果：

$$\boldsymbol{x}^{zx} \in \mathbb{R}^{b\times c\times H_z\times W_z}, \boldsymbol{x}^{xz} \in \mathbb{R}^{b\times c\times H_x\times W_x}$$

通过不断地堆叠 S-ECA 模块和 S-CFA 模块，进行n次上述 2 步计算后，输出最后一个 S-CFA 模块的$\boldsymbol{x}^{xz}$特征作为最终的相关运算结果。

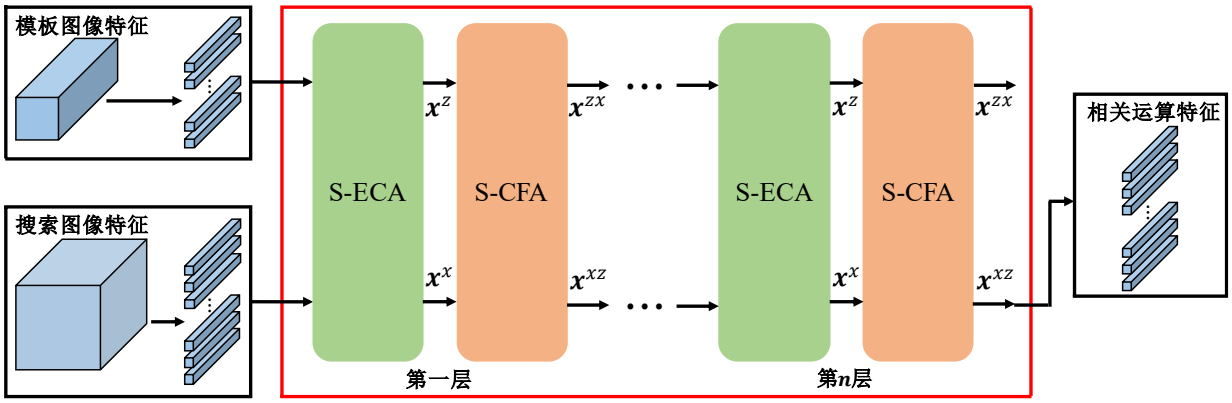

图 49　Transformer 相关运算模块

5.4 实验与分析

为了验证本章的孪生网络目标跟踪算法（称为“SiamSSA”）以及设计的相关运算模块的有效性，实验在全部 GOT-10k、YT-BB、COCO、DET 和 VID 五个训练集上进行模型训练，一共训练 1000 轮（epoch），并在 OTB100、GOT-10k 和 LaSOT 三个测试集上进行性能评估。因为过多地堆叠 S-ECA 模块与 S-CFA 模块将会大幅增加计算开销，

所以本章实验只按序堆叠一层 S-ECA 模块和一层 S-CFA 模块。

5.4.1 OTB100 测试集实验结果

本章设计的基于分流注意力多尺度特征融合的孪生网络跟踪算法在 OTB100 数据集上与近年各大顶级会议上发表的跟踪算法进行指标评比结果如表 12 与图 50 所示，其中表 12 和图 50 均展示成功率和精确率两项指标，各指标的解释详见 2.3 节。

表 12 SiamSSA 与各跟踪算法在 OTB100 测试集的指标评比结果

跟踪算法名称	OTB100 测试集	
	成功率（%）	精确率（%）
SAOT[42]	**71.4**	**92.6**
AutoMatch[38]	**71.4**	**92.6**
SiamGAT[16]	70.9	91.6
SiamRPN++[12]	69.6	92.3
AiATrack[43]	69.6	91.7
PrDimp50[44]	69.5	89.7
SiamSSA	**69.4**	**89.5**
TransT[20]	69.1	89.3
PGNet[15]	69.1	89.2
SPM[45]	68.7	89.9
Ocean[46]	68.4	92.0
DiMP[47]	68.4	-
SiamFC++[48]	68.3	-
ROAM[49]	68.1	90.8
SiamDW[11]	**67.4**	-
ATOM[50]	66.7	87.9
SiamRPN[51]	63.7	85.1

从图表中的结果可以看出，本章算法在 OTB100 数据集上的表现具有竞争力：该算法相较于基线算法 SiamDW[11]的成功率指标（表中蓝色标出）有了 2%的效果提升，相较于其它算法也均在成功率与精确率上有了部分提升，其中成功率和精确率均超过了只在单一尺度特征上进行融合的算法 TransT[20]，这表明本章设计的算法采用的多尺度特征融合技术是有效的。然而本章算法设计的相关运算模块只叠加了一层 S-ECA 模块与 S-CFA 模块，相关运算模块不够复杂，因此在 OTB100 数据集上的表现仍有待提升。

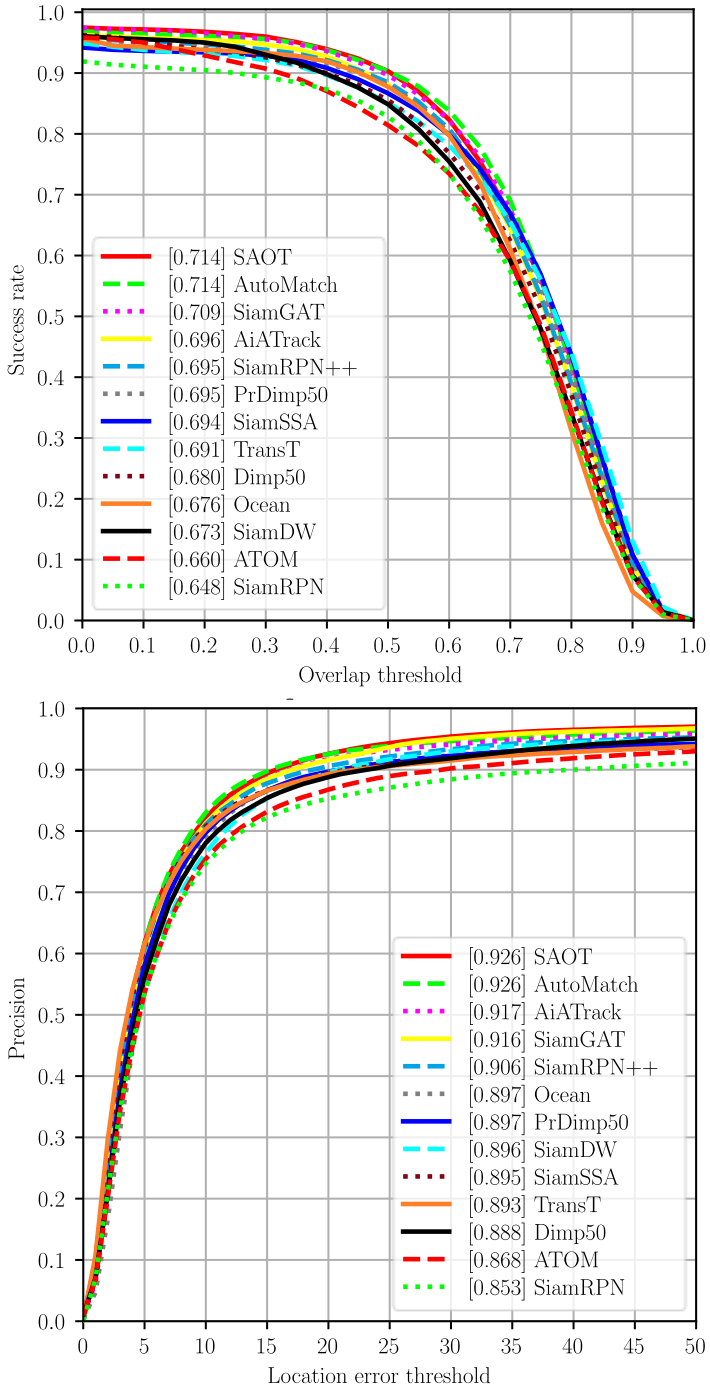

图 50　SiamSSA 在 OTB100 测试集的成功率图（上）与精确图（下）

5.4.2 GOT-10k 测试集实验结果

本章设计的基于相位感知注意力空间特征融合的孪生网络跟踪算法在 GOT-10k 数据集上与近年各大顶级会议上发表的跟踪算法进行指标评比结果如表 13 与图 51 所示。

表 13　SiamSSA 与各跟踪算法在 GOT-10k 测试集的指标评比结果

跟踪算法名称	GOT-10k 测试集		
	AO（%）	$SR_{0.5}$（%）	$SR_{0.75}$（%）

SiamSSA	**71.5**	**82.0**	**66.8**
AiATrack[43]	69.6	80.0	63.2
SPM[45]	68.7	89.9	
TransT[20]	67.1	76.8	60.9
AutoMatch[38]	65.2	76.6	54.3
SAOT[42]	64.0	74.9	-
PrDimp50[44]	63.4	73.8	54.3
SiamGAT[16]	62.7	74.3	48.8
Ocean[46]	61.1	72.1	-
DiMP[47]	61.1	71.7	49.2
SiamFC++[48]	59.5	69.5	47.9
ROAM[49]	46.5	53.2	23.6
SiamDW[11]	**41.6**	-	-
PGNet[15]	-	-	-
SiamRPN++[12]	-	-	-
ATOM[50]	-	-	-
SiamRPN[51]	-	-	-

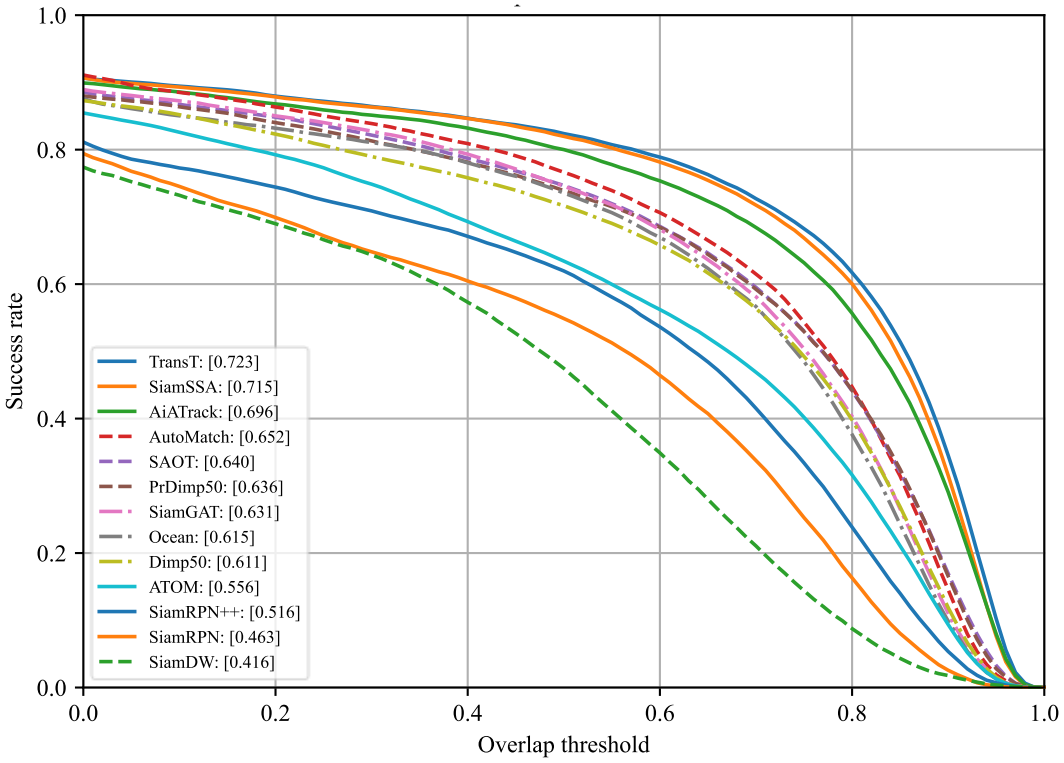

图 51 SiamSSA 在 GOT-10k 测试集的成功率图

从图表中的结果可以看出，本章算法在 GOT-10k 数据集上的表现具有很强的竞争

力：该算法相较于基线算法 SiamDW[11]的 AO 指标（表中蓝色标出）有了近 30%的效果提升，相较于其它算法也均在 AO、$SR_{0.5}$ 和 $SR_{0.75}$ 三项指标上有了较多地提升。本章的算法超过了只在单一尺度特征上进行相关运算的算法 TransT[20]，这表明本章设计的相关运算模块中采用的多尺度特征融合技术是有效的。

5.4.3 LaSOT 测试集实验结果

本章设计的基于相位感知注意力空间特征融合的孪生网络跟踪算法在 LaSOT 数据集上与近年各大顶级会议上发表的跟踪算法进行指标评比结果如表 14 与图 52 所示，其中表 14 和图 52 均展示成功率和精确率两项指标，各指标的解释详见 2.3 节。

表 14　SiamSSA 与各跟踪算法在 LaSOT 测试集的指标评比结果

跟踪算法名称	LaSOT 测试集	
	成功率（%）	精确率（%）
AiATrack[43]	**69.0**	**73.8**
TransT[20]	64.9	69.0
SiamSSA	**64.2**	**66.9**
SAOT[42]	61.6	70.8
PrDimp50[44]	59.8	-
AutoMatch[38]	58.3	59.9
DiMP[47]	56.9	-
Ocean[46]	56	56.6
SiamFC++[48]	54.4	-
PGNet[15]	53.1	60.5
SiamGAT[16]	53	53.9
ATOM[50]	51.4	50.5
SiamRPN++[12]	49.6	56.9
ROAM[49]	44.7	44.5
SiamDW[11]	**38.4**	-
SPM[45]	-	-
SiamRPN[51]	-	-

从图表中的结果可以看出，本章算法在 LaSOT 数据集上的表现有一些竞争力：该算法相较于基线算法 SiamDW[11]的成功率指标（表中蓝色标出）有了近 26%的提升，相较于其它算法也均在成功率与精确率上有了部分提升，并在只堆叠一层 S-ECA 模块和

一层 S-CFA 模块的情况下取得了和多层堆叠的 TransT[20]算法相近的性能表现。

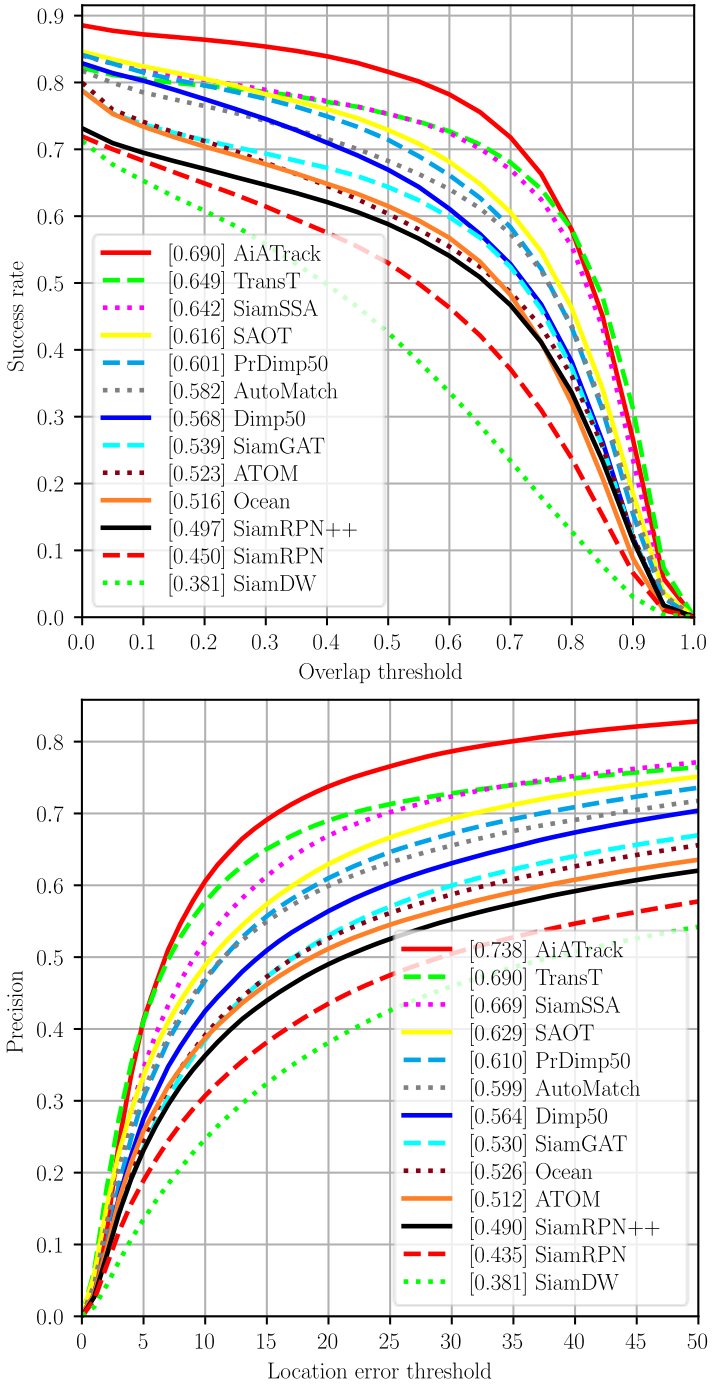

图 52 SiamSSA 在 LaSOT 测试集的成功率图（上）与精确图（下）

5.5 本章小结

本章提出了一种基于分流注意力多尺度特征融合的孪生网络跟踪算法，该算法对孪生网络目标跟踪中的相关运算技术进行改进：设计了一种基于 Transformer 的相关运算模块，该模块利用分流注意力提取单一尺度的查询算子和多个不同尺度的键算子及值算

子。内部设计了分流自主语境增强模块（S-ECA 模块）将上述算子进行增强表示，来学习模板图像特征以及搜索图像特征的多尺度特征增强；设计了分流交叉特征增强模块（S-CFA 模块）将上述算子进行融合表示，来学习模板图像特征和搜索图像特征的多尺度特征融合。

在实验部分，本章设计的基于分流注意力多尺度特征融合的孪生网络跟踪算法在 OTB100、GOT-10k 和 LaSOT 三个数据集上进行测试与对比。根据实验结果，本章设计的算法相较于基线算法都有了较高的性能提升，超过了大部分没有进行多尺度特征融合的算法，从实验的角度证明了利用在 Transformer 结构中进行多尺度特征融合的有效性。因为本章采用的 Transformer 相关运算模块暂时无法在地平线 X3 上进行硬件实现，所以第六章将不会对本章算法进行硬件实现。

第六章 基于地平线旭日 X3 平台的实时目标跟踪系统

前三章所述的三种算法分别从空间特征融合、通道特征融合和多尺度特征融合三个方面进行了相关运算技术的设计与改进，并在数据集上验证了算法的有效性。然而，数据集上的有效性并不直接决定算法在实际应用中的性能优良；同时，将算法落地尤其是在边缘设备上进行部署，将面临如计算资源有限、算法算子硬件不支持等难题。针对上述问题，本章以基于图注意力通道特征融合的孪生网络跟踪算法为例，实现在地平线旭日 X3 平台上的硬件部署，完成实时目标跟踪系统的搭建。

6.1 引言

根据前文所述，目标跟踪算法在航空航天领域、军事领域和民用领域都有广泛的应用。然而在很多实际应用场景下，必须采用边缘设备[71]进行算法的部署，而不是常见的个人电脑、服务器等等。如 2.1.2 节所述，孪生网络目标跟踪算法在线应用时，整个网络的参数将不再变化，这就要求设计的算法包含如下两个特点：

(1) 算法设计的网络尽量轻量化：

网络复杂程度的增大将会带来巨大的计算开销，对于计算资源有限的边缘设备而言将难以承担网络的运算。尤其是目标跟踪这种要求实时性地任务，网络的轻量化可以带来跟踪实时性地提升。

模型轻量化的方式可以分为：网络剪枝[72]（Network Pruning）、知识蒸馏[73]（Knowledge Distillation）、参数量化[72]（Parameter Quantization）和模型结构设计[74]（Architecture Design），本文不再对此展开叙述。

(2) 算法使用的算子的支持性、可替换性与可整合性：

本文第三、四、五三章设计的孪生网络由 PyTorch 上层深度学习框架实现，模型在服务器上进行硬件执行前会进行计算图优化，将设计的孪生网络分解为一个个算子，如二维卷积算子（torch.nn.Conv2d）、线性算子（torch.nn.Linear）、张量乘法算子（torch.matmul）等，这些算子在硬件实现时需要考虑它们的支持性、可替换性与可整合性。

①支持性：

以英伟达（Nvidia）显卡这一硬件环境为例，大部分上层框架会借助英伟达的通用并行计算架构 CUDA（Compute Unified Device Architecture）中的 cuDNN、cuBLAS 等

库针对具体硬件芯片特点来对算子进行优化实现。

然而对于大部分的边缘设备，它们使用与英伟达不同的硬件架构，具有不同的硬件特性，并不一定支持一些算子的优化。以旧版本的地平线 X3 开发板为例，RoiAlign 这一算子并不直接支持，这将限制一部分目标检测[39]和目标跟踪[38]的算法的部署。

②可替换性：

针对一些特殊的算子，虽然边缘设备的硬件架构并不直接支持，但是可以使用支持的算子进行替代实现，仍以地平线 X3 开发板为例，跟踪算法[75]中使用的 Swish 激活函数算子并不直接支持，但是该激活函数的公式如下：

$$\text{Swish}(x) = x \cdot \text{Sigmoid}(\beta x) \tag{6.1}$$

因此 Swish 激活函数算子可以使用Sigmoid算子以及 matmul 算子进行组合，完成替代实现。Swish 激活函数算子具有可替换性。

替换复杂的算子往往较为困难，这是因为往往需要数学上的等价推导。以跟踪算法[47]中出现的公式为例：

$$m_c + (1 - m_c) \cdot \frac{\partial \max(0, s)}{\partial s} \tag{6.2}$$

式 6.2 中$\max(0, s)$对s的导数定义如下：

$$\frac{\partial \max(0, s)}{\partial s} = \begin{cases} 1, & s > 0 \\ \frac{1}{2}, & s = 0 \\ 0, & s < 0 \end{cases} \tag{6.3}$$

因此可以做如下推导，式 6.2 等价于：

$$\frac{1 - m_c}{2} \cdot \text{sign}(s) + \frac{1 + m_c}{2} \tag{6.4}$$

故上述的导数计算可以使用符号函数sign算子进行实现，而sign算子是被地平线 X3 开发板所支持的。

③可整合性：

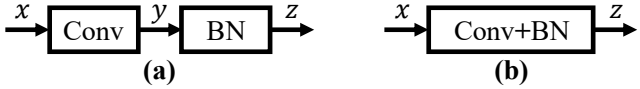

图 53　卷积算子与批正则化算子的整合

在模型的量化技术中，常将卷积算子（Conv）和批正则化算子（BatchNorm）整合成一个算子，以减少计算量。如图 53 所示，以一维卷积与一维批正则化为例：

记卷积算子的参数为权重$\boldsymbol{w}$与偏置$\boldsymbol{b}$，批正则化算子的参数为缩放系数γ和平移系数β，参数均在模型训练阶段学习获得，此时固定不变。

图 53（a）展示了卷积算子和批正则化算子分开先后进行操作：

卷积算子有：

$$\boldsymbol{y}=\begin{bmatrix}y_1\\y_2\\\vdots\\y_m\end{bmatrix}=\begin{bmatrix}w_{11}&w_{12}&\cdots&w_{1n}\\w_{21}&w_{22}&\cdots&w_{2n}\\\vdots&\vdots&\ddots&\vdots\\w_{m1}&w_{m2}&\cdots&w_{mn}\end{bmatrix}\begin{bmatrix}x_1\\x_2\\\vdots\\x_n\end{bmatrix}+\begin{bmatrix}b_1\\b_2\\\vdots\\b_m\end{bmatrix}=$$

$$[\boldsymbol{w}_1\quad\boldsymbol{w}_2\quad\cdots\quad\boldsymbol{w}_n]\begin{bmatrix}x_1\\x_2\\\vdots\\x_n\end{bmatrix}+\boldsymbol{b}=\sum_{i=1}^{n}(\boldsymbol{w}_i x_i)+\boldsymbol{b}\tag{6.5}$$

批正则化算子有：

$$\boldsymbol{z}=\gamma\frac{\boldsymbol{y}-\mu_y\boldsymbol{I}}{\sqrt{\sigma_y^2+\epsilon}}+\beta\boldsymbol{I}\tag{6.6}$$

其中，μ_y和σ_y^2为均值和方差，为了防止分母等于 0，ϵ设置为一个很小的正数，$\boldsymbol{I}$是单位向量。

将式 6.5 代入式 6.6 有：

$$\boldsymbol{z}=\gamma\frac{\sum_{i=1}^{n}(\boldsymbol{w}_i x_i)+\boldsymbol{b}-\mu_y\boldsymbol{I}}{\sqrt{\sigma_y^2+\epsilon}}+\beta\boldsymbol{I}=\frac{\gamma}{\sqrt{\sigma_y^2+\epsilon}}\sum_{i=1}^{n}(\boldsymbol{w}_i x_i)+\frac{\gamma}{\sqrt{\sigma_y^2+\epsilon}}\left(\boldsymbol{b}-\mu_y\boldsymbol{I}\right)+\beta\boldsymbol{I}$$

$$=\sum_{i=1}^{n}\left(\frac{\gamma}{\sqrt{\sigma_y^2+\epsilon}}\boldsymbol{w}_i x_i\right)+\frac{\gamma}{\sqrt{\sigma_y^2+\epsilon}}\boldsymbol{b}+\left(\beta-\frac{\gamma\mu_y}{\sqrt{\sigma_y^2+\epsilon}}\right)\boldsymbol{I}\tag{6.7}$$

如图 53（b）所示，卷积算子和批正则化算子可以整合成一个卷积算子实现，记整合成的卷积算子的参数为权重$\boldsymbol{w}'$与偏置$\boldsymbol{b}'$，则有：

$$\begin{cases}\boldsymbol{w}_i'=\dfrac{\gamma}{\sqrt{\sigma_y^2+\epsilon}}\boldsymbol{w}_i\\[2ex]\boldsymbol{b}'=\dfrac{\gamma}{\sqrt{\sigma_y^2+\epsilon}}\boldsymbol{b}+\left(\beta-\dfrac{\gamma\mu_y}{\sqrt{\sigma_y^2+\epsilon}}\right)\boldsymbol{I}\end{cases}\tag{6.8}$$

此时便将卷积算子和批正则化算子可以整合成一个新的卷积算子，简化了计算量。模型量化中还可以将卷积算子和 ReLU 激活函数算子进行整合，本文不再赘述。

图 54 展示了各种市面上常见的开发板，综合考虑价格、板上硬件资源及运算性能、硬件架构对算子支持程度，以及跟踪算法国产化的考虑，本文选用地平线旭日 X3 开发

板作为硬件平台进行基于图注意力通道特征融合的孪生网络跟踪算法的实现。

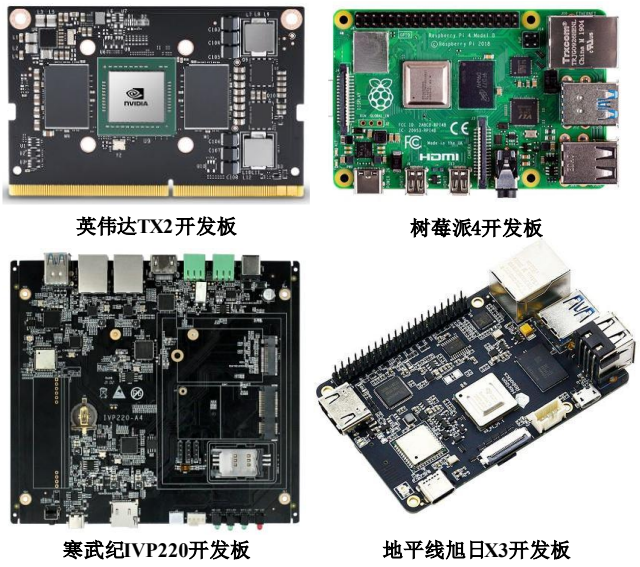

图 54　各种常见的开发板

6.2 模型部署

6.2.1 算子准备

在诸如 PyTorch、TensorFlow 等上层深度学习框架中使用图注意力算子往往借助于第三方代码库如 DGL（Deep Graph Library）库或者上层框架的拓展库如 PyG（PyTorch Geometric）库。虽然这些库可以帮助本文的工作在服务器上方便地进行图注意力算子的实现，然而这些第三方库或者拓展库并不直接支持国产边缘设备的部署。因此，如果要在边缘设备上实现图注意力算子，就必须对现有的图注意力算子进行替换实现。图结构上的图注意力算子及其邻接矩阵的示意图如图 55 所示。

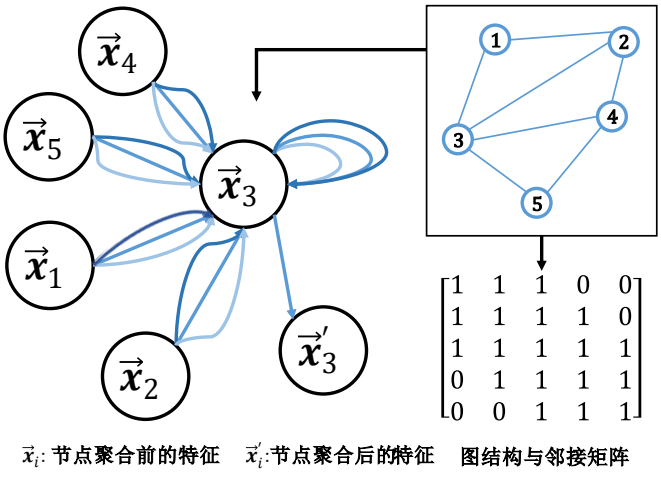

图 55　图结构上的图注意力算子及其邻接矩阵

图注意力算子的替换实现需要借助于图结构的邻接矩阵（Adjacency Matrix），记 4.2 章预先生成的图结构的邻接矩阵为$\boldsymbol{adj} \in \mathbb{R}^{c\times c}$，则图注意力算子分以下 2 步替换实现：

(1) 计算图注意力矩阵$\boldsymbol{a}$：

记图上的所有c个节点的特征为$\boldsymbol{x} \in \mathbb{R}^{c\times c'}$，则有：

$$\boldsymbol{x}' = \mathrm{Linear}_0(\boldsymbol{x}) \tag{6.9}$$

这里设置线性层Linear的输入特征维度为c'，输出特征维度为c''，则$\boldsymbol{x}' \in \mathbb{R}^{c\times c''}$，下面计算所有$c$个节点的注意力矩阵$\boldsymbol{atten} \in \mathbb{R}^{c\times c}$：

$$\begin{cases} \boldsymbol{e}_1 = \mathrm{Linear}_1(\boldsymbol{x}') \in \mathbb{R}^{c\times 1} \\ \boldsymbol{e}_2 = \mathrm{Linear}_2(\boldsymbol{x}') \in \mathbb{R}^{c\times 1} \end{cases} \tag{6.10}$$

这里设置线性层Linear的输入特征维度为c''，输出特征维度为 1。

所有c个节点的注意力矩阵$\boldsymbol{atten}$则为：

$$\boldsymbol{atten} = \boldsymbol{e}_1 \boldsymbol{e}_2^{\mathrm{T}} \tag{6.11}$$

然而图上的任一节点并非与其它所有节点相连，因此需要根据$\boldsymbol{adj}$筛选出注意力矩阵中有效的注意力值：

$$atten_{ij} = \begin{cases} atten_{ij}, & adj_{ij} = 1 \\ 0, & adj_{ij} = 0 \end{cases} \tag{6.12}$$

为了保证任一节点的邻居节点注意力权重之和为 1，再进行 Softmax 函数即可：

$$\boldsymbol{a} = \mathrm{Softmax}(\boldsymbol{atten}) \tag{6.13}$$

式 6.13 得到图注意力算子的注意力矩阵$\boldsymbol{a} \in \mathbb{R}^{c\times c}$。

(2) 计算基于图注意力卷积的节点聚合结果：

$$\hat{\boldsymbol{x}} = \mathrm{Linear}_4(\boldsymbol{a}\boldsymbol{x}') \tag{6.14}$$

其中，$\boldsymbol{a}\boldsymbol{x}' \in \mathbb{R}^{c\times c''}$，这里设置线性层Linear的输入特征维度为$c''$，输出特征维度为$c'$，因此$\hat{\boldsymbol{x}} \in \mathbb{R}^{c\times c'}$，保持和图注意力算子的输入$\boldsymbol{x}$一致的尺寸。

6.2.2 模型转换

将设计的模型部署在地平线旭日 X3 的开发板上需要将模型转换成特定的格式，地平线 AI 工具链是一种将模型量化的工具，它提供了将模型转换成板上可用的*.bin 格式的模型的指令接口，依次调用相应的接口便可以很快地完成量化，得到模型转换结果。该工具链的环境搭建本文不做介绍，下文依照图 56 简介模型转换流程：

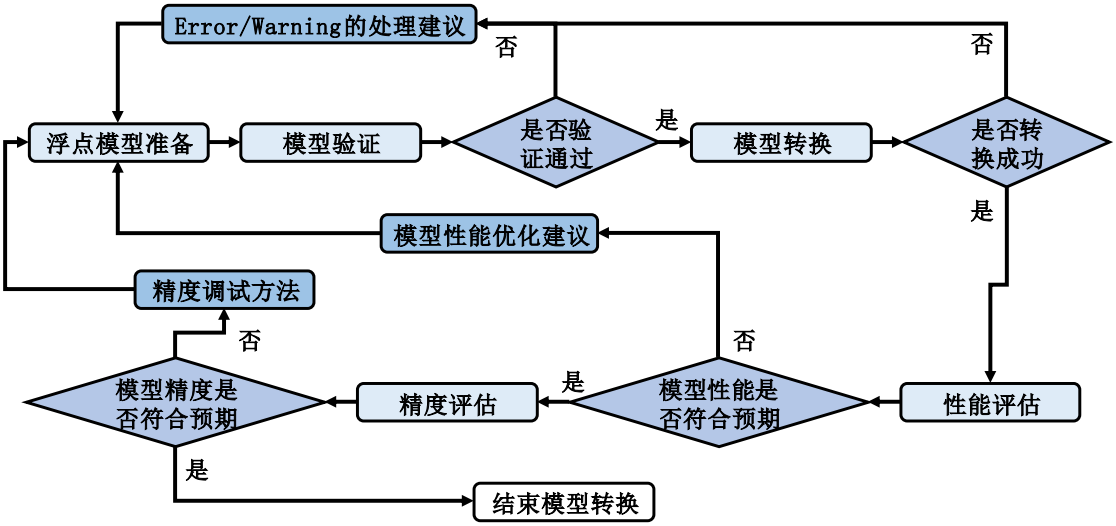

图 56　地平线工具链模型转换流程

(1) 浮点模型准备：

开放神经网络交换（Open Neural Network Exchange, ONNX）是一种用来表示深度学习模型的开放格式，整个工具链的输入并非是 PyTorch、TensorFlow 等上层深度学习框架中直接训练出的模型，而是需要将模型及其训练得到的参数一并转换成这种开放格式。地平线的工具链可以接受这种开放格式的浮点模型输入，并将其进行量化。

(2) 模型验证：

该步骤可以检验工具链得到的 ONNX 开放格式的模型是否符合量化工具链要求。对于不符合要求的情况，检查工具会明确给出不符合要求的具体算子信息，可以结合算子约束的说明将模型调整过来。检查指令使用 hb_mapper checker 指令。

在本文的工作中，工具链验证了 onnx 模型的合理性，检查了模型输入的名称及其对应的张量尺寸，其中 input1 为模板图像$\tilde{\boldsymbol{z}} \in \mathbb{R}^{1\times3\times127\times127}$，input2 为搜索图像特征$\tilde{\boldsymbol{x}} \in \mathbb{R}^{1\times3\times255\times255}$，input3 为模板图像中目标的边界框，为了配适 X3 平台将其尺寸设置属于$\mathbb{R}^{1\times4\times1\times1}$。工具链还对模型的算子等一并进行验证。

(3) 模型转换：

这一阶段将完成浮点模型到地平线混合异构模型的转换。工具链内部可以自行完成模型优化、量化和编译等关键步骤，保证模型能在 X3 芯片上高效运行。其中，量化环节需要提供标定（Calibration）文件，即模型训练过程中实际采用的输入示例，一般存成二进制*.bin 文件提供。

如表 15 所示，模型转换过程中会对所有算子进行评估，并选择将其置于 CPU 上进

行运算还是置于 BPU 上进行运算，同时给出了算子所在的计算图子图等信息。根据这些信息可以对模型进行调整，使得尽可能多的算子使用 BPU 进行加速运算。

表 15 SiamWAVE 跟踪算法部分算子转换结果示例

原先的算子类型	运行硬件	所在子图	转换后的算子类型	余弦相似度	阈值
Squeeze_0	CPU	--	Reshape	--	--
MaxPool_4	BPU	id(0)	HzQuantizedMaxPool	0.999793	1.665852
Conv_99	BPU	id(0)	HzSQuantizedConv	0.997664	0.923832
Slice_104	BPU	id(0)	Slice	0.997738	4.374025
Concat_230	CPU	--	Concat	1.000000	--
Gather_232	CPU	--	Gather	1.000000	--
Cast_234	CPU	--	Cast	1.000000	--
RoiAlign_237	CPU	--	RoiAlign	0.999614	--
LeakyRelu_239	BPU	id(2)	HzLeakyRelu	0.999833	2.173858
Expand_248	CPU	--	Expand	0.999833	2.173858
Tile_249	CPU	--	Tile	0.999833	--
Transpose_250	CPU	--	Transpose	--	--
Mul_276	BPU	id(3)	HzSQuantizedMul	0.990368	2.024365
Add_277	BPU	id(3)	HzSElementWiseAdd	0.991708	2.008431
UNIT_CONV_FOR_BatchNormalization_351	BPU	id(4)	HzSQuantizedConv	0.939072	3.193220
Cos_369	BPU	id(4)	HzLut	0.997089	6.115427
Sin_371	BPU	id(4)	HzLut	0.933484	6.115427
GlobalAveragePool_452_split_id_0	BPU	id(5)	HzQuantizedAveragePool	--	--
InstanceNormalization_515	CPU	--	InstanceNormalization	0.890202	--
UNIT_CONV_FOR_1121_TO_FUSE_RELU	BPU	id(14)	HzSQuantizedConv	--	--
Exp_546	BPU	id(14)	HzLut	0.844423	4.783722

(4) 性能评估与精度评估：

在应用部署之前，工具链可以验证模型的性能和精度是否达到应用要求，并输出如图 57 所示的硬件性能评估图。如果部分性能没有达到预期，可以进行相应地调优。

图 57 上图展示了模板图像的特征提取网络所在子图的理论性能评估结果，图 57 下

图展示了搜索图像的特征提取网络所在子图的理论性能评估结果。在 X3 芯片双核运行时，前者每秒传输帧数（Frames Per Second, FPS）为 92.3，延迟为 10.83 毫秒；后者 FPS 为 28.83，延迟为 34.68 毫秒。

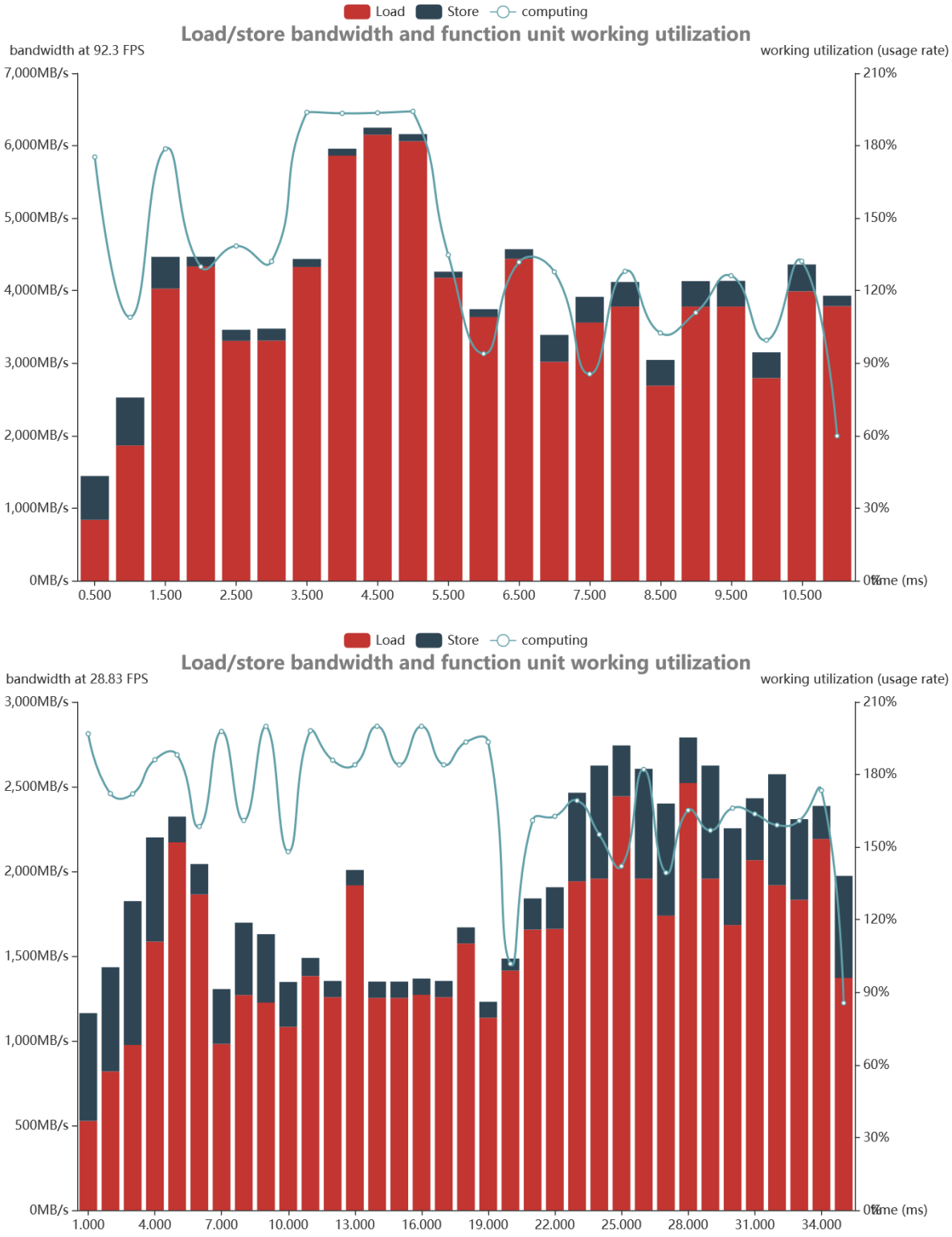

图 57　使用工具链进行模型转换

6.3 实验与实现

6.3.1 各章算法实验结果

为了验证本文第三、四、五章设计的孪生网络目标跟踪算法以及设计的相关运算模块的有效性，并对其进行组合实现，本章采用表 16 中的三种配置先在公开数据集上测试性能表现，再选择其中的一种配置搭建实时目标跟踪系统：

表 16 三种模型配置

模型配置	layer1+1+1	layer2+2+2	layer3+3+3

如图 58 所示，layer1+1+1 的配置是指并行堆叠第三章相关运算模块、第四章相关运算模块、第五章相关运算模块各一层；layer2+2+2 的配置是指在 layer1+1+1 的基础上再并行堆叠各章相关运算模块各一层；layer3+3+3 的配置同理堆叠三层。

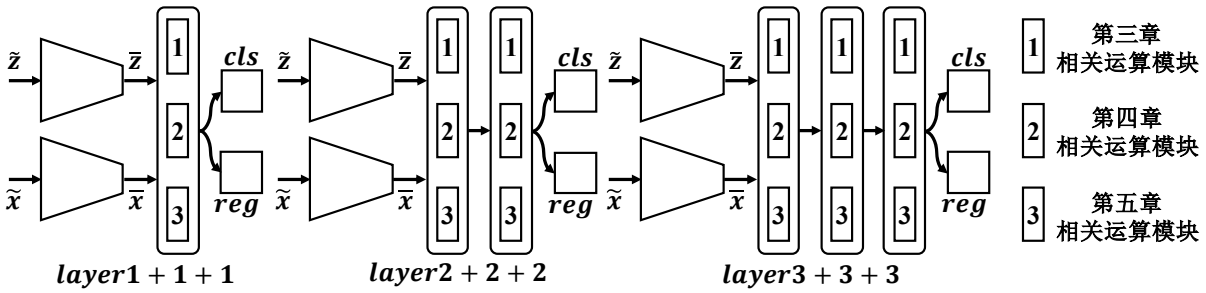

图 58 三种模型的配置示意图

三种配置的模型在全部 GOT-10k、YT-BB、COCO、DET 和 VID 五个训练集上进行模型训练，一共训练 32 轮（epoch），在前 10 轮冻结骨干网络使其参数在训练过程中不变，从第 11 轮开始对骨干网络和相关运算模块一并训练。

三种配置的模型以及本文所有涉及的算法在 OTB100、GOT-10k 和 LaSOT 三个测试集上进行性能评估的结果如表 17 所示：

表 17 本文所涉及的算法在三个数据集上的主要指标结果

算法	GOT-10k/AO	OTB100/成功率	LaSOT/成功率
layer3+3+3	**72.3**	**71.0**	**64.9**
layer2+2+2	**71.9**	**70.3**	**64.4**
layer1+1+1	**71.6**	**69.6**	**64.3**
SiamSSA	**71.5**	**69.4**	**64.2**
AiATrack[43]	69.6	69.6	69.0
SPM[45]	68.7	68.7	-

TransT[20]	67.1	69.1	64.9
SiamWAVE(layer3)	**66.1**	**67.7**	**54.7**
SiamGATPlus(layer3)	**65.8**	**70.3**	**54.6**
SiamWAVE(layer6)	**65.7**	**68.8**	**55.2**
SiamWAVE(layer5)	**65.4**	**67.5**	**53.5**
AutoMatch[38]	65.2	71.4	58.3
SiamWAVE(layer4)	**64.3**	**68.6**	**55.2**
SAOT[42]	64	71.4	61.6
PrDimp50[44]	63.4	69.5	59.8
SiamWAVE(layer2)	**63.2**	**67.1**	**52.7**
SiamGAT[16]	62.7	70.9	53
SiamGATPlus(layer1)	**62.2**	**69.9**	**54.4**
SiamWAVE(layer1)	**62.1**	**66.4**	**51.6**
SiamGATPlus(layer2)	**62.1**	**66.6**	**54.9**
Ocean[46]	61.1	68.4	56
DiMP[47]	61.1	68.4	56.9
SiamROI	**60.2**	**66.6**	**52.3**
SiamFC++[48]	59.5	68.3	54.4
ROAM[49]	46.5	68.1	44.7
SiamDW[11]	**41.6**	**67.4**	**38.4**
SiamRPN++[12]	-	69.6	49.6
SiamRPN[51]	-	63.7	-
PGNet[15]	-	69.1	53.1
ATOM[50]	-	66.7	51.4

表 17 中所有本文提出的算法的名称均被用不同于黑色的其它颜色标出：第三章的算法 SiamWAVE 及各配置（layer1-layer6）用绿色标出；第四章的算法 SiamGATPlus 及各配置（layer1-layer3）用玫红色标出；第五章的算法 SiamSSA 用紫色标出；本章的算法整合了前三章的算法并提出表 16 的三种配置，用橙色标出；SiamROI 算法表示只采用 3.2 节中的预相关运算模块做相关运算，等价于 SiamWAVE-layer0 的配置，亦等价于 SiamGATPlus-layer0 的配置，用黄色标出。表 17 中所有本文提出的算法的指标值均被用黑色加粗标出，各指标中的最高值被用红色加粗标出，基线算法的指标值被用蓝色加粗标出。

根据表 17 中的指标情况，第五章的基于分流注意力多尺度特征融合的孪生网络跟

踪算法优于第三章的基于相位感知注意力空间特征融合的孪生网络跟踪算法和第四章的基于图注意力通道特征融合的孪生网络跟踪算法。这说明基于 Transformer 的相关运算模块尽管带来了一些延迟，但是能够更好地融合模板图像特征与搜索图像特征，带来较高的跟踪性能。综合比较各种不同的配置，第三章的跟踪算法与第四章的跟踪算法在跟踪性能方面表现相近，但是第四章的算法由于静态随机图中节点数量不大导致各配置的表现不稳定，而第三章的算法以更小的计算开销得到稳定的跟踪性能。

考虑表 16 中的三种配置，表 17 说明将空间维度特征融合的多层感知机相关运算模块、通道维度特征融合的图神经网络相关运算模块、多尺度特征融合的 Transformer 相关运算模块进行整合，可以充分挖掘跟踪器的性能，实现跟踪效果的提升。综合考虑板上资源的限制、算子的适配以及跟踪算法在公开数据集上的性能表现，本章的硬件实现初步采用 layer3+3+3 的配置，并在搭建实时目标跟踪系统之后完成在飞机跟踪（航空航天应用）、导弹跟踪（军用）、车辆跟踪（民用）三个场景中的实际应用。

6.3.2 实时目标跟踪系统

记 layer3+3+3 的配置去除三层 Transformer 相关运算模块的配置为 layer3+3，只保留三层多层感知机相关运算模块的配置为 layer3，表 18 展示了运行在 1660ti 显卡上的三种配置在三个场景中的平均延迟（latency）。

表 18　1660ti 上三种配置在三个应用场景中的延迟

延迟	layer3	layer3+3	layer3+3+3
飞机跟踪	30.49 ms	43.27 ms	67.96 ms
导弹跟踪	27.80 ms	41.21 ms	66.17 ms
车辆跟踪	35.96 ms	42.68 ms	65.19 ms

考虑到表 18 中 Transformer 结构带来的高延迟，以及 Transformer 结构并不能很好地在地平线旭日 X3 平台上进行实现，因此去除其中的三层 Transformer 相关运算模块，以 layer3+3 的配置进行实时目标跟踪系统的搭建，搭建结果如图 59 所示。

如图 59 所示，基于地平线旭日 X3 平台的实时目标跟踪系统接收外部视频流的输入（MIPI 格式、SDI 格式等），并输送到图像预处理单元进行预处理（稳像、去模糊等），将处理后的图片存储在 DDR 子系统中。在接收外部指令开始进行跟踪后，系统从 DDR 中加载模型转换结果（*.bin 文件）以及第四章算法所需的预先生成的图结构，在 CPU 和 BPU 中组合运行跟踪器。跟踪器从 DDR 中逐帧读取预处理后的图片，并由分配在

CPU 上的算子和 BPU 上的算子进行多核并行计算，得到跟踪框，并通过后处理将新的跟踪框叠加在当前帧图片上，最后将图片重新编码成所需格式的视频流进行输出。

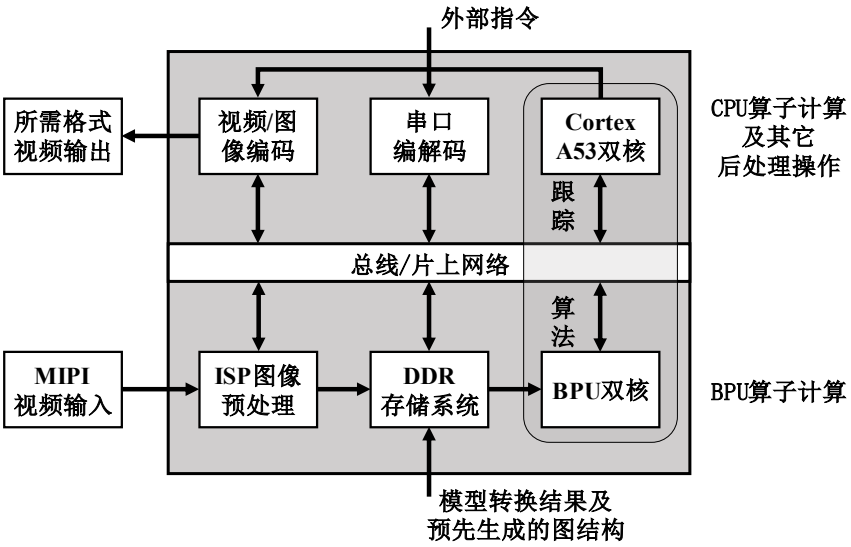

图 59　基于地平线旭日 X3 平台的实时目标跟踪系统

基于地平线旭日 X3 平台的实时目标跟踪系统在飞机跟踪、导弹跟踪、车辆跟踪三个场景中的实际应用结果如图 60 所示。

图 60 实时目标跟踪系统在实际场景中的应用结果

其中蓝色的框为 layer3 配置的结果（X3 上运行），绿色的框为 layer3+3 配置的结果（X3 上运行），红色的框为 layer3+3+3 配置的结果（1660ti 上运行），在 X3 上的延迟如表 19 所示。

表 19 X3 上三种配置在三个应用场景中的延迟

延迟	layer3	layer3+3	layer3+3+3
飞机跟踪	37.24 ms	51.25 ms	--
导弹跟踪	35.08 ms	50.11 ms	--
车辆跟踪	35.56 ms	50.12 ms	--

综合考虑图 60 中的跟踪结果与表 19 中的实际延迟，layer3+3 的配置在基于地平线旭日 X3 平台的实时目标跟踪系统上以可接受的延迟取得了良好的跟踪结果，在一些需要高实时性的简单跟踪场景中可以采用 layer3 的配置。

6.4 本章小结

本章将第三章及第四章的算法在地平线的旭日 X3 硬件平台上搭建了实时目标跟踪系统，其中针对第四章的图注意力卷积算子进行了算子替换。硬件部署过程中使用了地平线 AI 工具链将模型量化并转换成板端可以使用的模型格式，并给出了模型计算图的各子图在板端的理论硬件性能表现。实时目标跟踪系统在飞机跟踪、导弹跟踪及车辆跟踪三个实际应用场景进行使用并取得良好的跟踪效果。

结论与展望

1 工作总结

本课题针对孪生网络目标跟踪算法中相关运算的关键技术进行了深入探究，设计了一种基于相位感知注意力机制的多层感知机相关运算模块、一种基于图注意力机制的图神经网络相关运算模块、一种基于分流注意力机制的 Transformer 相关运算模块，并综合考虑实际的硬件条件将前两种设计在地平线旭日 X3 平台上进行了硬件实现：

本文具体研究成果的总结如下：

(1) 设计了基于相位感知注意力空间特征融合的孪生网络跟踪算法：

该算法采用预相关运算模块和基于多层感知机的相关运算模块进行组合，对空间维度特征的相关运算进行改进。其中基于多层感知机的相关运算模块将图像或图像特征视为一种“波”，并采用不含显式注意力算子的相位感知注意力机制，自适应地调整图像 token 之间的权重并进行融合。算法的不同配置在 OTB100、GOT-10k 和 LaSOT 数据集上进行性能评估，评估结果显示该算法取得了良好的跟踪性能表现。这种略去注意力算子的设计以及简单的多层感知机结构降低了网络的复杂度与计算开销，因此被采用在地平线旭日 X3 平台上进行硬件实现。

(2) 设计了基于图注意力通道特征融合的孪生网络跟踪算法：

该算法采用预相关运算模块和基于图神经网络的相关运算模块进行组合，对通道维度特征的相关运算进行尝试。其中基于图神经网络的相关运算模块先采用 Watts-Strogatz 随机图生成算法生成一个图结构，再采用该图结构建模通道维度特征的相关性，图注意力卷积对该图结构与通道维度特征进行融合。算法的不同配置在 OTB100、GOT-10k 和 LaSOT 数据集上进行性能评估，评估结果显示该算法取得了良好的跟踪性能表现。这种使用图结构建模相关性并采用图神经网络进行相关运算的设计在进行硬件实现时的困难具有典型性，因此被采用在地平线旭日 X3 平台上进行硬件实现。

(3) 设计了基于分流注意力多尺度特征融合的孪生网络跟踪算法：

该算法采用基于 Transformer 的相关运算模块对多尺度特征的相关运算进行设计。该模块内部设计了分流自主语境增强模块 S-ECA 和分流交叉特征增强模块 S-CFA，前者提取单一尺度的查询算子和多个不同尺度的键算子及值算子并采用自注意力机制进

行增强表示，后者提取这些算子并采用互注意力机制进行特征融合。基于 Transformer 的相关运算模块由 S-ECA 和 S-CFA 交替堆叠搭建，实现 Transformer 结构内部不同尺度的特征融合。算法在 OTB100、GOT-10k 和 LaSOT 数据集上进行性能评估，评估结果显示该算法取得了良好的跟踪性能表现。这种使用复杂的 Transformer 结构进行相关运算的设计无法在地平线旭日 X3 平台上进行硬件实现。

(4) 实现了基于地平线旭日 X3 平台的实时目标跟踪系统：

使用地平线 AI 工具链将基于相位感知注意力空间特征融合的孪生网络跟踪算法及基于图注意力通道特征融合的孪生网络跟踪算法转换成板端可以使用的模型格式，其中图注意力卷积算子进行了算子替换。模型计算图的各子图在板端的理论硬件性能表现良好。实时目标跟踪系统成功在航天器跟踪、军事目标跟踪及监控行人跟踪三个实际应用场景进行使用。

2 工作展望

尽管本课题针对孪生网络目标跟踪算法中相关运算的关键技术进行了深入探究与改进设计，但是仍有提升空间。根据本课题所研究的内容的未来发展趋势，工作展望总结如下：

(1) 现有的相关运算技术计算$\bar{\boldsymbol{z}}$和$\bar{\boldsymbol{x}}$的相关性，其中$\bar{\boldsymbol{z}}$是模板图像经过特征提取网络后的特征，$\bar{\boldsymbol{x}}$是搜索图像经过特征提取网络后的特征。这种相关性的计算只在高维抽象的特征上进行，忽略了低维细节的特征对于相关性计算的影响。一些最新的工作[76,77]开始将相关运算技术与特征提取网络进行合并，在特征提取的各个阶段（低维细节的特征→高维抽象的特征）都进行相关运算，形成一种一边进行特征提取一边进行相关运算的结构。本课题的工作可以进一步借鉴这种结构，将空间维度特征相关运算、通道维度特征相关运算、多尺度特征相关运算都与特征提取网络进行合并，理论上可以取得更优的跟踪性能表现。

(2) 实现孪生网络目标跟踪算法的落地必须将网络轻量化，虽然目前一些工作已经做出尝试，如采用神经架构搜索获得轻量化的模型结构[75]、采用已有的轻量化特征提取网络替换现有的复杂特征提取网络[78]、直接对 Transformer 之类的复杂网络结构进行轻量化设计[79]，但是如何兼顾边缘设备上的跟踪速度与跟踪精度仍旧有待探究。

参考文献

[1] 李玺, 查宇飞, 张天柱, 崔振, 左旺孟, 侯志强, 卢湖川, 王菡子. 深度学习的目标跟踪算法综述[J]. 中国图象图形学报, 2019, 24(12): 2057-2080

[2] Pandey D., Wairya S., Sharma M., et al. An Approach for Object Tracking, Categorization, and Autopilot Guidance for Passive Homing Missiles[J]. Aerospace Systems, 2022: 1-14

[3] Jha S., Seo C., Yang E., et al. Real Time Object Detection and Trackingsystem for Video Surveillance System[J]. Multimedia Tools and Applications, 2021, 80(3): 3981-3996

[4] Arashpour M., Ngo T., Li H.. Scene Understanding in Construction and Buildings Using Image Processing Methods: A Comprehensive Review and A Case Study[J]. Journal of Building Engineering, 2021, 33: 101672

[5] Wu S., Zhang K., Li S., et al. Joint Feature Embedding Learning and Correlation Filters for Aircraft Tracking with Infrared Imagery[J]. Neurocomputing, 2021, 450: 104-118

[6] 王少博, 张成, 苏迪, 冀瑞静. 基于改进YOLOv3和核相关滤波算法的旋转弹目标探测算法[J]. 兵工学报, 2022, 43(05): 1032-1045

[7] 段玉瑶, 马丽, 刘刚. 猪舍场景下的生猪目标跟踪和行为检测方法研究[J]. 农业机械学报, 2015, 46(S1): 187-193

[8] 方澄, 路稳, 姬菁颖, 宋玉蒙, 梁斐菲, 罗志伟. 基于外观相似性更新的相关滤波跟踪算法[J]. 系统工程与电子技术, 2022, 44(01): 117-126

[9] Javed S., Danelljan M., Khan F S., et al. Visual Object Tracking with Discriminative Filters and Siamese Networks: A Survey and Outlook[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022: 1-20

[10] Kristan M., Matas J., Leonardis A., et al. The Ninth Visual Object Tracking VOT2021 Challenge Results[A]. IEEE International Conference on Computer Vision Workshops[C]. Los Alamitos, CA: IEEE Computer Society, 2021: 2711-2738

[11] Zhang Z., Peng H.. Deeper and Wider Siamese Networks for Real-Time Visual Tracking[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2019: 4586-4595

[12] Li B., Wu W., Wang Q., et al. SiamRPN plus plus: Evolution of Siamese Visual Tracking

with Very Deep Networks[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2019: 4277-4286

[13] Bertinetto L., Valmadre J., Henriques J F., et al. Fully-Convolutional Siamese Networks for Object Tracking[A]. European Conference on Computer Vision[C]. Berlin: Springer, 2016: 850-865

[14] Han W., Dong X., Khan F S., et al. Learning to Fuse Asymmetric Feature Maps in Siamese Trackers[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2021: 16565-16575

[15] Liao B., Wang C., Wang Y., et al. PG-Net: Pixel to Global Matching Network for Visual Tracking[A]. European Conference on Computer Vision[C]. Berlin: Springer, 2020: 429-444

[16] Guo D., Shao Y., Cui Y., et al. Graph Attention Tracking[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2021: 9538-9547

[17] Velickovic P., Cucurull G., Casanova A., et al. Graph Attention Networks[A]. International Conference on Learning Representations[C]. OpenReview: ICLR, 2018: 1-12

[18] Vaswani A., Shazeer N., Parmar N., et al. Attention is All You Need[J]. Advances in neural information processing systems, 2017, 30: 1-11

[19] Dosovitskiy A., Beyer L., Kolesnikov A., et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale[A]. International Conference on Learning Representations[C]. OpenReview: ICLR, 2021: 1-21

[20] Chen X., Yan B., Zhu J., et al. Transformer Tracking[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2021: 8122-8131

[21] Tang Y., Han K., Guo J., et al. An Image Patch is a Wave: Quantum Inspired Vision MLP[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2022: 10935-10944

[22] Xu B., Yin H.. Graph Convolutional Networks in Feature Space for Image Deblurring and Super-resolution[A]. IEEE International Joint Conference on Neural Networks[C]. New York: IEEE, 2021: 1-8

[23] Ren S., Zhou D., He S., et al. Shunted Self-Attention via Multi-Scale Token Aggregation[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2022: 10853-10862

[24] Fan H., Lin L., Yang F., et al. Lasot: A High-quality Benchmark for Large-scale Single Object Tracking[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2019: 5369-5378

[25] Huang L., Zhao X., Huang K.. GOT-10k: A Large High-Diversity Benchmark for Generic Object Tracking in the Wild[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2019, 43(5): 1562-1577

[26] Real E., Shlens J., Mazzocchi S., et al. YouTube-BoundingBoxes: A Large High-Precision Human-Annotated Data Set for Object Detection in Video[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2017: 7464-7473

[27] Lin T Y., Maire M., Belongie S., et al. Microsoft COCO: Common Objects in Context[A]. European Conference on Computer Vision[C]. Berlin: Springer, 2014: 740-755

[28] Deng J., Dong W., Socher R., et al. ImageNet: A Large-Scale Hierarchical Image Database[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2009: 248-255

[29] Wu Y., Lim J., Yang M H.. Online Object Tracking: A Benchmark[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2013: 2411-2418

[30] Guo M H., Liu Z N., Mu T J., et al. Beyond Self-Attention: External Attention Using Two Linear Layers for Visual Tasks[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022: 1-13

[31] Tolstikhin I O., Houlsby N., Kolesnikov A., et al. Mlp-mixer: An All-mlp Architecture

for Vision[J]. Advances in Neural Information Processing Systems, 2021, 34: 24261-24272

[32] Touvron H., Bojanowski P., Caron M., et al. ResMLP: Feedforward Networks for Image Classification With Data-Efficient Training[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022: 1-9

[33] Liu H., Dai Z., So D., et al. Pay Attention to Mlps[J]. Advances in Neural Information Processing Systems, 2021, 34: 9204-9215

[34] Yu T., Li X., Cai Y., et al. S2-mlp: Spatial-shift Mlp Architecture for Vision[A]. IEEE Winter Conference on Applications of Computer Vision[C]. New York: IEEE, 2022: 297-306

[35] Hou Q., Jiang Z., Yuan L., et al. Vision Permutator: A Permutable MLP-Like Architecture for Visual Recognition[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022: 1-8

[36] Lian D., Yu Z., Sun X., et al. AS-MLP: An Axial Shifted MLP Architecture for Vision[A]. International Conference on Learning Representations[C]. OpenReview: ICLR, 2021: 1-19

[37] Chen S., Xie E., Chongjian G E., et al. CycleMLP: A MLP-like Architecture for Dense Prediction[A]. International Conference on Learning Representations[C]. OpenReview: ICLR, 2021: 1-21

[38] Zhang Z., Liu Y., Wang X., et al. Learn to Match: Automatic Matching Network Design for Visual Tracking[A]. IEEE International Conference on Computer Vision[C]. New York: IEEE, 2021: 13319-13328

[39] He K., Gkioxari G., Dollár P., et al. Mask R-CNN[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020, 42(2): 386-397

[40] He K., Zhang X., Ren S., et al. Deep Residual Learning for Image Recognition[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. New York: IEEE, 2016: 770-778

[41] Sandler M., Howard A., Zhu M., et al. MobileNetV2: Inverted Residuals and Linear Bottlenecks[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. New

York: IEEE, 2018: 4510-4520

[42] Zhou Z., Pei W., Li X., et al. Saliency-Associated Object Tracking[A]. IEEE International Conference on Computer Vision[C]. New York: IEEE, 2021: 9846-9855

[43] Gao S., Zhou C., Ma C., et al. AiATrack: Attention in Attention for Transformer Visual Tracking[A]. European Conference on Computer Vision[C]. Berlin: Springer, 2022: 146-164

[44] Danelljan M., Gool L V., Timofte R.. Probabilistic Regression for Visual Tracking[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. New York: IEEE, 2020: 7181-7190

[45] Wang G., Luo C., Xiong Z., et al. SPM-Tracker: Series-Parallel Matching for Real-Time Visual Object Tracking[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2019: 3638-3647

[46] Zhang Z., Peng H., Fu J., et al. Ocean: Object-aware Anchor-free Tracking[A]. European Conference on Computer Vision[C]. Berlin: Springer, 2020: 771-787

[47] Bhat G., Danelljan M., Gool L V., et al. Learning Discriminative Model Prediction for Tracking[A]. IEEE International Conference on Computer Vision[C]. New York: IEEE, 2019: 6181-6190

[48] Xu Y., Wang Z., Li Z., et al. SiamFC plus plus: Towards Robust and Accurate Visual Tracking with Target Estimation Guidelines[A]. AAAI Conference on Artificial Intelligence[C]. Palo Alto, CA: AAAI, 2020, 34: 12549-12556

[49] Yang T., Xu P., Hu R., et al. ROAM: Recurrently Optimizing Tracking Model[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. New York: IEEE, 2020: 6717-6726

[50] Danelljan M., Bhat G., Khan F S., et al. ATOM: Accurate Tracking by Overlap Maximization[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2019: 4655-4664

[51] Li B., Yan J., Wu W., et al. High Performance Visual Tracking with Siamese Region Proposal Network[A]. IEEE Conference on Computer Vision and Pattern Recognition[C].

New York: IEEE, 2018: 8971-8980

[52] Wu S., Sun F., Zhang W., et al. Graph Neural Networks in Recommender Systems: A Survey[J]. ACM Computing Surveys, 2020: 1-37

[53] Shi W., Rajkumar R.. Point-GNN: Graph Neural Network for 3D Object Detection in a Point Cloud[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. New York: IEEE, 2020: 1708-1716

[54] Wieder O., Kohlbacher S., Kuenemann M., et al. A Compact Review of Molecular Property Prediction with Graph Neural Networks[J]. Drug Discovery Today: Technologies, 2020, 37: 1-12

[55] Liu Y., Wang W., Hu Y., et al. Multi-Agent Game Abstraction via Graph Attention Neural Network[A]. AAAI Conference on Artificial Intelligence[C]. Palo Alto, CA: AAAI, 2020: 7211-7218

[56] Han K., Wang Y., Guo J., et al. Vision GNN: An Image is Worth Graph of Nodes[J]. Advances in Neural Information Processing Systems, 2022

[57] Watts D J., Strogatz S H.. Collective Dynamics of 'Small-World' Networks[J]. nature, 1998, 393(6684): 440-442

[58] Erdős P., Rényi A.. On the Evolution of Random Graphs[J]. Publ. Math. Inst. Hung. Acad. Sci, 1960, 5(1): 17-60

[59] Barrat A., Weigt M.. On the Properties of Small-World Network Models[J]. The European Physical Journal B-Condensed Matter and Complex Systems, 2000, 13(3): 547-560

[60] Al-Anzi B., Arpp P., Gerges S., et al. Experimental and Computational Analysis of A Large Protein Network that Controls Fat Storage Reveals the Design Principles of A Signaling Network[J]. PLoS computational biology, 2015, 11(5): e1004264

[61] Wu H., Xu J., Wang J., et al. Autoformer: Decomposition Transformers with Auto-correlation for Long-term Series Forecasting[J]. Advances in Neural Information Processing Systems, 2021, 34: 22419-22430

[62] Rong Y., Bian Y., Xu T., et al. Self-supervised Graph Transformer on Large-scale Molecular Data[J]. Advances in Neural Information Processing Systems, 2020, 33: 12559-

12571

[63] Moritz N., Hori T., Le J.. Streaming Automatic Speech Recognition with the Transformer Model[A]. International Conference on Acoustics Speech and Signal Processing[C]. New York: IEEE, 2020: 6074-6078

[64] Touvron H., Cord M., Douze M., et al. Training Data-efficient Image Transformers & Distillation through Attention[A]. Proceedings of Machine Learning Research[C]. San Diego, CA: JMLR, 2021: 7358-7367

[65] Wang W., Xie E., Li X., et al. Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions[A]. IEEE International Conference on Computer Vision[C]. New York: IEEE, 2021: 548-558

[66] Yuan L., Chen Y., Wang T., et al. Tokens-to-Token ViT: Training Vision Transformers from Scratch on ImageNet[A]. IEEE International Conference on Computer Vision[C]. New York: IEEE, 2021: 538-547

[67] Han K., Xiao A., Wu E., et al. Transformer in Transformer[J]. Advances in Neural Information Processing Systems, 2021, 34: 15908-15919

[68] Wu K., Peng H., Chen M., et al. Rethinking and Improving Relative Position Encoding for Vision Transformer[A]. IEEE International Conference on Computer Vision[C]. New York: IEEE, 2021: 10033-10041

[69] Chen B., Li P., Li C., et al. GLiT: Neural Architecture Search for Global and Local Image Transformer[A]. IEEE International Conference on Computer Vision[C]. New York: IEEE, 2021: 12-21

[70] Liu Z., Lin Y., Cao Y., et al. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows[A]. IEEE International Conference on Computer Vision[C]. New York: IEEE, 2021: 10012-10022

[71] 乔德文，郭松涛，何静，朱永东. 边缘智能：研究进展及挑战[J]. 无线电通信技术, 2022, 48(01): 34-45

[72] Liang T., Glossner J., Wang L., et al. Pruning and Quantization for Deep Neural Network Acceleration: A Survey[J]. Neurocomputing, 2021, 461: 370-403

[73] Gou J., Yu B., Maybank S J., et al. Knowledge Distillation: A Survey[J]. International Journal of Computer Vision, 2021, 129(6): 1789-1819

[74] Ren P., Xiao Y., Chang X., et al. A Comprehensive Survey of Neural Architecture Search: Challenges and Solutions[J]. ACM Computing Surveys, 2021, 54(4): 1-34

[75] Yan B., Peng H., Wu K., et al. LightTrack: Finding Lightweight Neural Networks for Object Tracking via One-shot Architecture Search[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2021: 15175-15184

[76] Cui Y., Jiang C., Wang L., et al. MixFormer: End-to-End Tracking with Iterative Mixed Attention[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Los Alamitos, CA: IEEE Computer Society, 2022: 13608-13618

[77] Ye B., Chang H., Ma B., et al. Joint Feature Learning and Relation Modeling for Tracking: A One-Stream Framework[J]. arXiv preprint arXiv: 2203.11991, 2022

[78] Borsuk V., Vei R., Kupyn O., et al. FEAR: Fast, Efficient, Accurate and Robust Visual Tracker[J]. arXiv preprint arXiv: 2112.07957, 2021

[79] Blatter P., Kanakis M., Danelljan M., et al. Efficient Visual Tracking with Exemplar Transformers[J]. arXiv preprint arXiv: 2112.09686, 2021

攻读硕士学位期间取得的研究成果

1 攻读硕士学位期间申请的国家发明专利

[1] 张弘, **沈天琦**, 杨一帆. 一种应用于机动平台的嵌入式孪生网络实时跟踪方法[P]. 中国专利: 202111127652.8, 2021-12-28 (已受理)

2 攻读硕士学位期间参与的主要科研项目

[1] 某型号飞机吊舱目标检测跟踪系统，院所项目，2020 年 11 月至 2021 年 6 月，参与人：算法研究及实现。

[2] 某型号红外空地目标跟踪系统，院所项目，2021 年 11 月至 2022 年 6 月，参与人：软硬件设计、算法研究及实现。

致谢

感谢父母及其他亲人。

感谢张弘老师、袁丁老师、陈浩师兄、杨一帆师兄、李旭亮师兄、张磊师兄、张泽宇师兄、李亚伟师兄、吴昱颉师兄、伍凌帆师兄、李岩师兄、冯亚春师兄、闫超奇师兄、邢万里师兄、吕毅轩师兄、张恺师兄、刘翰阳师兄、韩杨师兄、曹一荻师兄、张鹏师兄、宋剑波同学、万家旭同学、郭威威同学、赵存飞同学、邓阳艳同学、孙运龙师弟、曹书豪师弟、周炫锋师弟、刘源师弟、娄亚鑫师弟、王清可师弟、李源师弟、陈栋华师弟、刘家炜师弟、张思哲师妹、林华靠师弟、芮思语师妹等。

感谢张子众同学、刘一平同学、冯家齐同学、刘少华同学、陈乐宇同学、李亦非同学、于锐渤同学、王瑜同学、晁永越同学、刘文涵同学、何永宁同学。

感谢鲁冰新学长、赵培炎学长、张宇振同学、赖嵩同学、张屹垚同学。

感谢范登平老师、李兆玺学长、曾诚学长、袁伟民学长、王一诺学长、吕书畅学长、许泽宇学长、高威学长、沈维发学长、邓鑫亮学长、柳致远、李禹陪。

感谢张宇哲、张成、谢玥、李晓涵同学、赵欣玥同学、陈琳同学、李佳学长、罗绮。

感谢刘孟亭同学、李定哲同学、许欣然同学、王晨丹同学、张可昕学姐、王淑玉学姐、郑玉学姐、江铃学姐、李亚玲学妹、林子涵学妹、毛忆南学弟、朱杰学弟、吴威龙学弟，王艳辅导员、梁炫烨辅导员、赵凯辅导员、李国刊辅导员、王立东辅导员、李明智辅导员等。

感谢安宁、邱舒翰、孙增鹏、王金柯、高峰、徐国晟、姜瑞滢，聂其杨、彭盼、陈章、衣梅圣等。

感谢所有北航的老师。

感谢我的猫和狗，虽然它们还没有出现，希望未来能遇见。

感谢黄璐女士，一路走来互相磨合和扶持不易，但是自从你在我的人生中出现，我便看到了前进的方向，希望以后继续一起努力。

感谢我自己。