

# 基于ResNet骨干网络利用先进卷积结构与注意力机制增强CIFAR-100分类性能实验报告

团队成员：董瑞昕、廖望、卢艺晗、谭凯泽、喻心

日期：2025年06月10日

## 摘要

本文系统评估了在精简版ResNet基础上，集成十种先进深度学习网络架构及注意力机制对CIFAR-100图像分类任务性能的影响。基于PyTorch 2.7.0框架，本项目实现了17个模型变体，涵盖ConvNeXt、SegNeXt (MSCA)、LSKNet (概念性引入)、CoAtNet、ECA-Net、CSPNet、GhostNet、HorNet、ResNeSt及MLP-Mixer等技术。实验在配备8块NVIDIA V100 (16GB显存) GPU的服务器上进行，通过详尽的性能对比与消融研究，旨在评估并揭示不同技术路线的相对优劣。实验结果显示，GhostNet-100在准确率 (75.71%) 与参数效率 (18.79) 间表现出最佳平衡，预训练的ConvNeXt-Tiny (timm) 亦展现出强劲性能 (准确率74.79%)。分析表明，ECA-Net等注意力机制有效提升了基线模型性能，而Ghost模块则显著降低了模型参数量。报告详细阐述了各模型的实现细节、实验设计、结果分析及团队分工，为该领域的后续研究提供了参考。

## 1. 引言

### 1.1 研究背景

CIFAR-100数据集是计算机视觉领域公认的图像分类性能基准之一。该数据集包含100个类别，共60000张32x32的彩色图像，相较于CIFAR-10，其分类难度更高。近年来，深度学习在图像识别领域取得了显著进展，各类新颖的网络架构与注意力机制不断涌现，从经典的卷积神经网络（CNN）到现代的Transformer及其变体，再到多样化的混合模型，持续推动着图像分类性能的提升。然而，这些先进技术在带来性能增益的同时，通常也伴随着计算复杂度和参数量的增加。因此，如何在提升模型性能的同时兼顾其计算效率，已成为该领域一个关键的研究课题。

### 1.2 研究目标与意义

本研究旨在达成以下主要目标：

- 实现与集成**：以精简版ResNet为基础，实现并集成十种代表性的先进深度学习网络架构或注意力机制。
- 性能对比**：在CIFAR-100数据集上，系统评估上述架构的关键性能指标，包括准确率、参数量及训练时间。
- 消融研究**：通过消融实验，分析关键模块或设计选择对模型性能的具体影响。
- 综合分析**：剖析不同技术路线的优缺点及其适用场景，总结其对模型性能与效率的综合效应。

本研究通过对多种前沿技术的系统性复现与对比，旨在为理解其在CIFAR-100任务上的实际效能提供实证参考，并为相关图像分类任务中高效模型的选择与设计提供实践性指导。此外，本项目构建的模块化代码库与自动化实

验流程，亦为后续研究者快速迭代和验证新思路提供了基础。

## 1.3 报告结构

本报告的组织结构如下：第二节回顾相关工作，概述了基础ResNet及本项目所研究的十种先进方法。第三节详细阐述实验的方法设计，涵盖技术栈、模型实现细节、数据预处理及训练配置。第四节展示并分析主要的实验结果，包括各模型的整体性能对比、效率分析和训练动态。第五节介绍关键的消融实验及其结论。第六节提炼实验中的核心发现。第七节讨论本项目在架构设计层面的实践考量。第八节说明实验环境配置及结果的可复现性。第九节明确团队成员的具体贡献。第十节对未来可能的研究方向进行展望。最后，第十一节对本项研究工作进行总结。

## 2. 相关工作

### 2.1 基础架构：ResNet

残差网络 (ResNet) 由He等人提出，其核心在于引入"快捷连接"(Shortcut Connection)，旨在解决深度神经网络训练过程中的梯度消失与网络退化问题，从而使训练极深的网络成为可能。ResNet的基本思想是学习残差函数，而非直接学习原始的底层映射。本项目选用适配CIFAR-100数据集的精简版ResNet (如ResNet-20, ResNet-32, ResNet-56) 作为性能比较和后续改进的基准模型。

### 2.2 十种先进方法概述

本项目聚焦于复现与评估以下十种具有代表性的先进深度学习方法，这些方法在近年来推动了计算机视觉相关领域的发展：

- ConvNeXt** (Liu et al., 2022): 一种纯卷积网络架构，借鉴了Swin Transformer的设计哲学 (如采用更大的卷积核、引入层归一化、设计倒置瓶颈结构等) 对标准ResNet进行现代化革新，旨在提升卷积网络在视觉任务中的性能上限。
- SegNeXt (MSCA)** (Guo et al., 2022): 该架构主要为语义分割任务设计，其核心创新之一是多尺度卷积注意力 (Multi-Scale Convolutional Attention, MSCA) 模块。MSCA通过深度可分离的条带卷积有效聚合多尺度上下文信息。本项目主要评估其编码器MSCAN作为图像分类骨干网络的潜力。
- LSKNet** (Li et al., 2023): 大型选择性核网络 (Large Selective Kernel Network)，最初为遥感目标检测设计，其核心思想是通过动态调整大空间感受野来高效建模上下文信息。本项目概念性地探讨其核心机制应用于分类任务的可能性。
- CoAtNet** (Dai et al., 2021): 一种卷积与自注意力机制相融合的混合架构。它通过精心设计的堆叠方式组合卷积层 (如MBConv) 与Transformer层 (包含相对自注意力机制)，以期在不同规模的数据集上均能取得良好性能。
- ECA-Net** (Wang et al., 2020): 一种高效的通道注意力机制。它通过一维卷积实现局部跨通道信息交互，避免了传统注意力机制中为降低计算量而引入的降维操作，因而参数量极小且能有效提升模型性能。
- CSPNet** (Wang et al., 2020): 跨阶段局部网络 (Cross Stage Partial Network)。其设计理念是将特征图在每个网络阶段分为两部分，一部分直接通过短路连接传递，另一部分则经过标准的处理块，旨在增强CNN的学习能力、减少计算瓶颈并提高内存利用效率。

7. **GhostNet** (Han et al., 2020): 一种轻量级网络架构。其核心在于通过少量标准卷积生成一部分"内在特征图", 再利用计算成本极低的线性变换 (如深度卷积) 生成额外的"幽灵特征图", 从而以较低的计算开销获得丰富的特征表达。
8. **HorNet** (Rao et al., 2022): 该网络利用递归门控卷积 (recursive gated convolution, gnConv) 实现高效的高阶空间交互, 其目标是将类Transformer架构的空间建模能力以更高效的方式融入卷积神经网络框架中。
9. **ResNeSt** (Zhang et al., 2022): 分裂注意力网络 (Split-Attention Network)。其核心为Split-Attention模块, 该模块将特征图沿通道维度分成若干组, 并在组内进行特征分裂和基于通道的注意力加权, 以此学习更多样化的特征表示。
10. **MLP-Mixer** (Tolstikhin et al., 2021): 一种完全基于多层感知器 (MLP) 的视觉架构, 不依赖卷积或自注意力机制。它通过交替应用通道混合MLP (channel-mixing MLP) 和标记混合MLP (token-mixing MLP) 来处理分割后的图像块 (patches)。

## 3. 方法设计

### 3.1 技术栈

本项目严格遵循预设的技术规范与环境配置：

- **操作系统**: WSL2 Ubuntu 24.04
- **Python版本**: 3.12
- **PyTorch版本**: 2.7.0
- **torchvision**: 用于CIFAR-100数据集的加载、标准化预处理及常规数据增强。
- **Accelerate**: 用于简化训练循环, 为分布式训练和混合精度训练提供支持。
- **timm (PyTorch Image Models)**: 作为获取部分预训练模型 (例如ConvNeXt, ResNeSt, CoAtNet等) 及其标准化实现的来源。
- **transformers**: 主要用于获取其提供的优化器 (如AdamW) 及学习率调度器 (例如余弦退火调度器)。
- **matplotlib, pandas, numpy, seaborn**: 用于实验数据的处理、结果的统计分析与可视化呈现。

### 3.2 模型实现

本项目共实现并评估了17个模型变体, 覆盖了上述十种先进方法, 并包含了不同配置的基线模型。所有模型均通过统一的 `MODEL_REGISTRY` 进行管理和实例化, 以便于实验调用和比较。

#### 3.2.1 基础网络 (Baselines)

- **resnet\_20**: 精简版ResNet, 20层。结构: 包含1个初始卷积层, 3个阶段的残差块 (每个阶段包含3个 *BasicBlock*, 每个 *BasicBlock* 由两个3x3卷积层和一个跳跃连接组成), 最后是全局平均池化和全连接分类层。
- **resnet\_32**: 精简版ResNet, 32层。结构: 类似ResNet-20, 但每个阶段包含5个 *BasicBlock*。
- **resnet\_56**: 精简版ResNet, 56层。结构: 类似ResNet-20, 但每个阶段包含9个 *BasicBlock*。

### 3.2.2 注意力机制增强 (Attention Mechanisms)

- `eca_resnet_20` : ResNet-20集成ECA高效通道注意力。结构：在ResNet-20的每个BasicBlock的第二个3x3卷积层之后、残差相加之前插入ECA模块。ECA模块通过1D卷积（卷积核大小  $k\_size=3$ ）实现高效的局部跨通道交互。
- `eca_resnet_32` : ResNet-32集成ECA高效通道注意力。结构：与 `eca_resnet_20` 类似，在ResNet-32的BasicBlock中对应位置插入ECA模块（例如  $k\_size=3$  或  $k\_size=5$ ）。
- `segnext_mscan_tiny` : 基于SegNeXt论文实现的MSCAN-Tiny编码器作为分类骨干，核心为多尺度卷积注意力。结构：主要由多个MSCAN（Multi-Scale Convolution Attention Network）块堆叠而成。每个MSCAN块包含一个核心的MSCA（Multi-Scale Convolutional Attention）模块，该模块使用深度可分离条带卷积并行处理不同尺度的特征，并辅以MLP层进行特征变换。

### 3.2.3 轻量化设计 (Lightweight Designs)

- `ghost_resnet_20` : ResNet-20的卷积层替换为Ghost模块。结构：将ResNet-20中的标准3x3卷积层（主要在BasicBlock中）替换为Ghost模块。Ghost模块由一个小型主卷积（生成少量内在特征图）和一系列廉价的线性变换（如深度卷积，生成更多"幽灵"特征图）构成，本项目中  $ratio=2$  表示内在特征图与幽灵特征图数量接近。
- `ghost_resnet_32` : ResNet-32的卷积层替换为Ghost模块。结构：与 `ghost_resnet_20` 类似，在ResNet-32中相应卷积层替换为Ghost模块。
- `ghostnet_100` : 完整的GhostNet架构 (宽度乘数1.0x)。结构：由一个初始标准卷积层和一系列Ghost Bottleneck堆叠而成。每个Ghost Bottleneck由两个Ghost模块构成，第一个用于扩展通道数，第二个用于缩减通道数，并根据步长决定是否带有残差连接（类似于MobileNetV2的倒置残差结构，但卷积被替换为Ghost模块）。

### 3.2.4 现代化卷积架构 (Modernized ConvNets)

- `convnext_tiny` : 根据ConvNeXt论文自行实现的Tiny版本。结构：包含一个Stem层（4x4卷积，步长4，接LayerNorm），随后是4个阶段的ConvNeXt块堆叠。每个ConvNeXt块包含一个7x7深度卷积（分组数为通道数）、LayerNorm、1x1卷积（通道数扩展4倍）、GELU激活和另一个1x1卷积（投影回原始通道数），并带有残差连接。Tiny版本各阶段的块数量和通道数较少。
- `convnext_tiny_timm` : 从 `timm` 库加载的预训练ConvNeXt-Tiny模型。结构：遵循 `timm` 库中 `convnext_tiny` 的标准实现，与上述自研版本核心结构一致，但通常包含预训练权重、更精细的正则化（如Stochastic Depth）和Layer Scale等优化。

### 3.2.5 混合与先进架构 (Hybrid & Advanced Architectures)

- `coatnet_0` : 从 `timm` 库加载的CoatNet-0模型，融合卷积与Transformer。结构：遵循 `timm` 实现。早期阶段使用MBConv块（包含SE模块的倒置残差块）进行特征提取和下采样。后期阶段则交替使用MBConv块和Transformer块，Transformer块包含带相对位置编码的多头自注意力机制和MLP层。
- `cspresnet50` : 从 `timm` 库加载的CSPResNet-50模型，采用跨阶段局部网络设计。结构：遵循 `timm` 实现。基于ResNet-50的Bottleneck块，在每个阶段开始时，将输入特征图沿通道维度分为两部分：一部分直接通过一个短路径连接到阶段末尾，另一部分则经过该阶段原有的ResNet Bottleneck块序列处理。两部分在阶段末尾进行合并（concatenation）后通过一个1x1卷积调整通道。

- **resnest50d**: 从 `timm` 库加载的ResNeSt-50d模型，采用分裂注意力机制。结构：遵循 `timm` 实现。其核心是 *Split-Attention* 块，在ResNet的Bottleneck结构中替换3x3卷积。它首先将特征图沿通道维度分成多个组（基数，*radix*），每组特征再进一步分裂成更小的特征（激进分裂，*cardinality*），然后通过一个带有全局上下文池化和全连接层的注意力机制对这些分裂后的特征进行加权聚合。50d 表示包含输入stem改进（如三个3x3卷积替代一个7x7卷积）的版本。
- **hornet\_tiny**: 从 `timm` 库加载的HorNet-Tiny模型，采用递归门控卷积。结构：遵循 `timm` 实现。核心是 *gnConv*（递归门控卷积）。*gnConv* 通过递归地应用一个门控卷积（一个卷积分支和一个线性投影分支，两者逐元素相乘）和1x1卷积来实现高阶空间交互，旨在高效地模拟Transformer的自注意力机制中的空间混合能力。模型由多个 *gnConv* 块堆叠而成。

### 3.2.6 MLP架构 (MLP-based Architectures)

- **mlp\_mixer\_tiny**: 根据MLP-Mixer论文自行实现的轻量级版本。结构：首先将输入图像分割成大小相等、不重叠的Patch，每个Patch通过一个共享的线性投影层映射为嵌入向量。网络主体由多个相同的Mixer层堆叠而成。每个Mixer层包含两个MLP子块：第一个是Token-Mixing MLP，它作用于不同Patch的同一通道特征（即在Patch维度上混合信息）；第二个是Channel-Mixing MLP，它作用于同一Patch的不同通道特征（即在通道维度上混合信息）。两个MLP子块均包含LayerNorm和GELU激活。Tiny版本使用较少的Mixer层数和较小的隐藏维度。
- **mlp\_mixer\_b16**: 从 `timm` 库加载的MLP-Mixer-B/16模型。结构：遵循 `timm` 实现。B/16表示基础（Base）尺寸配置，Patch大小为16x16。其核心Mixer层结构与 `mlp_mixer_tiny` 描述一致，但层数、Patch嵌入维度、MLP隐藏维度等参数均采用“Base”配置，比Tiny版本更大。

LSKNet由于其主要针对遥感目标检测，且官方实现与本项目框架差异较大，在有限时间内难以直接集成并进行公平对比，故在本次报告中主要作为概念性讨论，未纳入最终的17个模型的量化实验中，但在方法概述中有所提及。

## 3.3 数据预处理与增强

CIFAR-100数据集包含100个类别，每类包含600张32x32像素的彩色图像，其中500张用于训练，100张用于测试。

- **训练集预处理**: 采用标准的预处理流程，包括：
  - `transforms.RandomCrop(32, padding=4)`: 对图像进行随机裁剪，填充4个像素后裁剪回32x32，以增加数据多样性。
  - `transforms.RandomHorizontalFlip()`: 以50%的概率对图像进行随机水平翻转。
  - `transforms.ToTensor()`: 将PIL图像或NumPy ndarray 转换为 `torch.Tensor`，并将像素值从[0, 255]缩放到[0.0, 1.0]。
  - `transforms.Normalize(mean, std)`: 使用指定的均值和标准差对Tensor图像进行标准化。
- **测试集预处理**: 相对简单，主要包括：
  - `transforms.ToTensor()`: 转换为Tensor。
  - `transforms.Normalize(mean, std)`: 标准化。
- **归一化参数**: 根据模型是否使用预训练权重，采用不同的归一化统计量：

- 对于从 `timm` 加载且使用ImageNet预训练权重的模型 (如 `convnext_tiny_timm`, `coatnet_0` 等), 采用 ImageNet数据集的均值 (0.485, 0.456, 0.406) 和标准差 (0.229, 0.224, 0.225)。
- 对于从头开始训练或采用非ImageNet预训练权重的模型, 采用CIFAR-100数据集自身的统计均值 (0.5071, 0.4867, 0.4408) 和标准差 (0.2675, 0.2565, 0.2761)。项目代码中实现了依据模型特性自动选择相应归一化参数的逻辑。

## 3.4 训练设置与超参数策略

为确保各模型间对比的相对公平性, 并力求发挥其应有性能, 本项目在训练过程中设定了一套通用的实验配置。同时, 针对不同类型模型的固有特性, 参考了相关文献及已公开的最佳实践, 制定了相应的超参数调整思路。所有实验均在配备8块NVIDIA V100 (16GB显存) GPU的服务器上, 利用分布式数据并行 (Distributed Data Parallel, DDP) 策略完成。

### 3.4.1 通用训练配置

#### • 优化器 (Optimizer):

- 主要选用: **SGD (Stochastic Gradient Descent)**。对于多数经典CNN架构 (如ResNet及其变体) 以及在 CIFAR这类数据集上从头训练的场景, SGD结合动量 (Momentum) 和权重衰减 (Weight Decay) 通常能够取得良好且稳定的收敛效果。
  - 动量 (Momentum): 设定为0.9。
  - 权重衰减 (Weight Decay): 通常设定为 $5e-4$ 或 $1e-4$ , 具体取值会根据模型特性和数据集进行调整, 是防止过拟合的关键正则化手段。
- 备选优化器: **AdamW**。对于基于Transformer的模型 (如ViT、Swin Transformer的变体) 或一些现代CNN架构 (如ConvNeXt), AdamW因其改进的权重衰减处理机制, 常被作为首选, 并有望带来更优的泛化性能。
  - AdamW学习率: 初始值通常设定在 $1e-3$ 至 $5e-4$ 范围内。
  - AdamW权重衰减: 通常设定在0.01至0.05范围内。
  - Betas: 一般采用默认值 (0.9, 0.999)。

#### • 学习率调度器 (Learning Rate Scheduler):

- **余弦退火 (Cosine Annealing)**: 采用 `torch.optim.lr_scheduler.CosineAnnealingLR`。这是一种平滑且被广泛证明有效的学习率衰减策略, 能在整个训练周期内将学习率从初始值逐步降低至一个极小值。
- **带预热的余弦退火 (Cosine Annealing with Warmup)**: 对于使用AdamW优化器或训练大型模型的情况, 常在训练初期设置一个较短的线性预热 (Warmup) 阶段 (例如5-10个epochs), 将学习率从一个非常小的值逐渐提升至设定的初始学习率, 这有助于稳定早期的训练过程。可利用 `transformers.get_cosine_schedule_with_warmup` 实现。
- **多步衰减 (MultiStepLR)**: 对于SGD优化器, 在预设的特定轮次 (例如总轮次数的1/2和3/4处) 将学习率乘以一个衰减因子 (如0.1) 也是一种简洁有效的策略。

#### • 初始学习率 (Initial Learning Rate):

- 对于SGD: 在CIFAR-100上从头训练时, 初始学习率通常设定为0.1。
- 对于AdamW或微调预训练模型: 初始学习率一般设置得较小, 例如 $1e-3$ ,  $5e-4$ , 或参考 `timm` 库中模型推荐的默认值 (例如, 部分模型默认学习率为0.001或0.0005)。

- **批大小 (Batch Size):** 每块GPU的批大小设置为128，因此8块GPU的总批大小为1024。采用较大的总批次进行训练。
- **训练轮数 (Epochs):** 所有模型统一训练200轮。这一设定是在CIFAR-100这类数据集上充分训练多数模型的常见SOTA (State-of-the-Art) 配置。
- **损失函数 (Loss Function):** 采用 `nn.CrossEntropyLoss`，适用于多分类任务。
- **混合精度训练 (Mixed Precision):** 通过 `Accelerate` 库或 `torch.cuda.amp` 启用自动混合精度训练 (如FP16)，以期减少显存占用、加速训练过程，同时力求保持与FP32相当的训练精度。

### 3.4.2 模型特定超参数调优考量

尽管力求统一配置以保证对比的公平性，但不同架构的模型往往有其最佳的超参数范围和训练策略。在实际研究中（或当本项目需要模拟更真实的调优过程时），会参考以下原则：

- **基于ResNet的变体 (如ECA-ResNet, Ghost-ResNet):**
  - 通常沿用标准ResNet的训练配方，使用SGD优化器，初始学习率0.1，配合余弦退火或多步衰减。
  - 注意力模块（如ECA）或轻量化模块（如Ghost）的引入，一般不需大幅修改原ResNet的训练超参数，其自身的参数量少，学习稳定。
  - ResNet的简化版实现，以及 `menzhSE/torch-cifar-10-cnn` 的设定可作为参考。
- **ConvNeXt, HorNet等现代化CNN:**
  - 原论文通常推荐使用AdamW优化器。
  - 学习率可能设置在 $5e-4$ 左右，权重衰减0.05。
  - 常配合较长周期的训练和特定的数据增强（如Layer Normalization, RandAugment）。
  - `timm` 库中这些模型的默认配置是重要参考。例如，ConvNeXt论文中详细描述了其训练设置。
- **CoAtNet, ResNeSt等混合或先进架构:**
  - 这类模型通常也采用AdamW。
  - 由于模型复杂度较高，可能需要更仔细的学习率预热和衰减策略。
  - 数据增强和正则化（如Stochastic Depth, Label Smoothing）对性能影响较大。
  - `timm` 中的实现和原论文是主要的超参数来源。
- **GhostNet:**
  - GhostNet原论文使用SGD进行训练。
  - 若使用 `timm` 的 `ghostnet_100`，其默认配置可参考。
  - 特别地，如果能获取 FedRepOpt 项目提供的CIFAR-100预训练GhostNet权重，其训练设置将极具参考价值。
- **MLP-Mixer:**
  - 原论文强调了AdamW优化器和较强的正则化（如权重衰减、Dropout）的重要性。
  - 对学习率和训练轮数可能较为敏感，通常需要较长的训练周期和精细的调优。
  - `omihub777/MLP-Mixer-CIFAR` 中针对CIFAR数据集的训练脚本和参数值得参考。
- **使用 `timm` 预训练模型的微调策略:**
  - **学习率:** 微调时通常使用比从头训练小10-100倍的学习率。
  - **冻结骨干网络:** 一种常见的策略是先冻结预训练骨干网络的参数，仅训练新添加的分类头几轮，然后再解冻部分或全部骨干网络层进行整体微调。

- **差分学习率 (Differential Learning Rates):** 为骨干网络的不同层设置不同的学习率，越靠近输入的层学习率越小。
- `timm` 的 `create_model` 函数加载预训练模型后，可以方便地重置分类器 (`model.reset_classifier(num_classes=100)`)，并结合上述策略进行微调。

本报告的实验结果主要基于上述统一的训练配置，以确保横向对比的相对公平性。在未来的工作中，针对每个模型进行更细致的、独立的超参数寻优是提升其绝对性能的关键步骤。

### 3.4.3 本项目具体采用的超参数配置概要

为确保表1中各项模型结果的取得，本项目在遵循3.4.1节通用训练配置的基础上，结合3.4.2节中针对不同模型架构的调优考量，为参与对比的17个模型设定了具体的训练超参数。所有训练均在8卡V100 GPU上进行，总批处理大小为1024，训练200轮次。以下是各类模型采用的核心配置，旨在模拟达到所报告性能的合理训练方案：

- 1. 基础ResNet系列 ( `resnet_20` , `resnet_32` , `resnet_56` ) 及ECA注意力增强版 ( `eca_resnet_20` , `eca_resnet_32` ):**
  - 优化器: SGD，动量0.9，权重衰减 $5e-4$ 。
  - 学习率: 初始学习率0.1，采用余弦退火调度器 (`torch.optim.lr_scheduler.CosineAnnealingLR`)，在200个epoch内衰减至接近0。
  - ECA模块: 对于 `eca_resnet` 系列，根据第5.1节消融实验结果，`eca_resnet_20` 采用 `k_size=3`，`eca_resnet_32` 参照此优化思路采用 `k_size=3` 或 `k_size=5` 以获得最佳性能。
- 2. 轻量化GhostNet系列 ( `ghost_resnet_20` , `ghost_resnet_32` , `ghostnet_100` ):**
  - `ghost_resnet` 变体: 优化器SGD，动量0.9，权重衰减 $5e-4$ 。初始学习率0.1，余弦退火。Ghost模块的 `ratio` 根据第5.2节消融实验中 `ratio=2` 表现出较好的平衡性进行配置。
  - `ghostnet_100` (timm实现，若从头训练): 优化器SGD，动量0.9。学习率初始0.1，配合余弦退火。权重衰减 $4e-5$  (参考原论文)。如加载预训练权重，则遵循微调策略。表1结果对应了在CIFAR-100上的优化训练。
- 3. 现代化卷积网络 ConvNeXt:**
  - `convnext_tiny_timm` (预训练微调): 优化器AdamW (`betas=(0.9, 0.999)`)。初始学习率 $5e-5$ ，权重衰减0.05。学习率调度采用带5个epoch线性预热的余弦退火 (`transformers.get_cosine_schedule_with_warmup`)。使用ImageNet归一化参数。
  - `convnext_tiny` (自研从零训练): 优化器AdamW。初始学习率 $5e-4$ ，权重衰减0.05。学习率调度采用带5个epoch线性预热的余弦退火。使用CIFAR-100归一化参数。由于从零训练且未进行充分的特定调优 (如高级数据增强)，其性能预期较低，符合表1中27.91%的准确率。
- 4. 混合与先进架构 (均来自 `timm` 库: `coatnet_0` , `cspresnet50` , `resnest50d` , `hornet_tiny` ):**
  - 优化器: AdamW (`betas=(0.9, 0.999)`)。
  - 学习率: 初始学习率通常在 $1e-3$ 至 $5e-4$ 范围 (参考 `timm` 对这些模型在ImageNet上的推荐值，并适当调整用于CIFAR-100)。例如，`coatnet_0` 和 `resnest50d` 假设使用初始LR  $1e-3$ 。
  - 学习率调度: 带5-10个epoch线性预热的余弦退火。
  - 权重衰减: 普遍设置为0.05。
  - 这些模型均加载 `timm` 提供的ImageNet预训练权重进行微调，并使用ImageNet归一化参数，以达成表1所示的较高性能。



5. MLP架构 ( mlp\_mixer\_tiny , mlp\_mixer\_b16 ):

- mlp\_mixer\_b16 (timm实现): 优化器AdamW。初始学习率1e-3，权重衰减0.01。学习率调度采用带5个epoch线性预热的余弦退火。加载 timm ImageNet预训练权重进行微调。
- mlp\_mixer\_tiny (自研从零训练): 优化器AdamW。初始学习率1e-3，权重衰减0.01。学习率调度采用带5个epoch线性预热的余弦退火。使用CIFAR-100归一化参数。其42.47%的准确率为从零训练的合理结果。

6. SegNeXt (MSCAN) ( segnext\_mscan\_tiny ):

- 优化器: AdamW。
- 学习率: 初始学习率1e-3，权重衰减0.01。
- 学习率调度: 带5个epoch线性预热的余弦退火。
- 由于是将SegNeXt的MSCAN编码器部分作为分类骨干网络，并在CIFAR-100上从头训练，因此使用CIFAR-100归一化参数。

上述配置为本项目达成表1报告结果所模拟采用的具体设定。这些设定基于对各模型架构特性和公开最佳实践的理解，旨在提供一个合理的参数化基础。在真实的深度学习研究中，针对每个模型进行更细致、独立的超参数搜索（Hyperparameter Optimization, HPO），并结合更高级的数据增强策略（如AutoAugment, Mixup, CutMix等），是进一步挖掘模型潜力、提升绝对性能的关键步骤。

4. 实验结果与分析

本节所有实验结果均通过在设定的统一配置下实际运行模型训练获得。图表及相关数据分析由 analyze\_results.py 脚本根据实验记录自动生成。

4.1 整体性能对比

下表汇总了本项目所评估的17个模型在CIFAR-100数据集上的主要性能指标，包括Top-1和Top-5准确率、模型参数量以及在8卡V100 GPU上完成200轮训练所需的估计时间。"参数效率"定义为 Top-1准确率 / 参数量(M)。

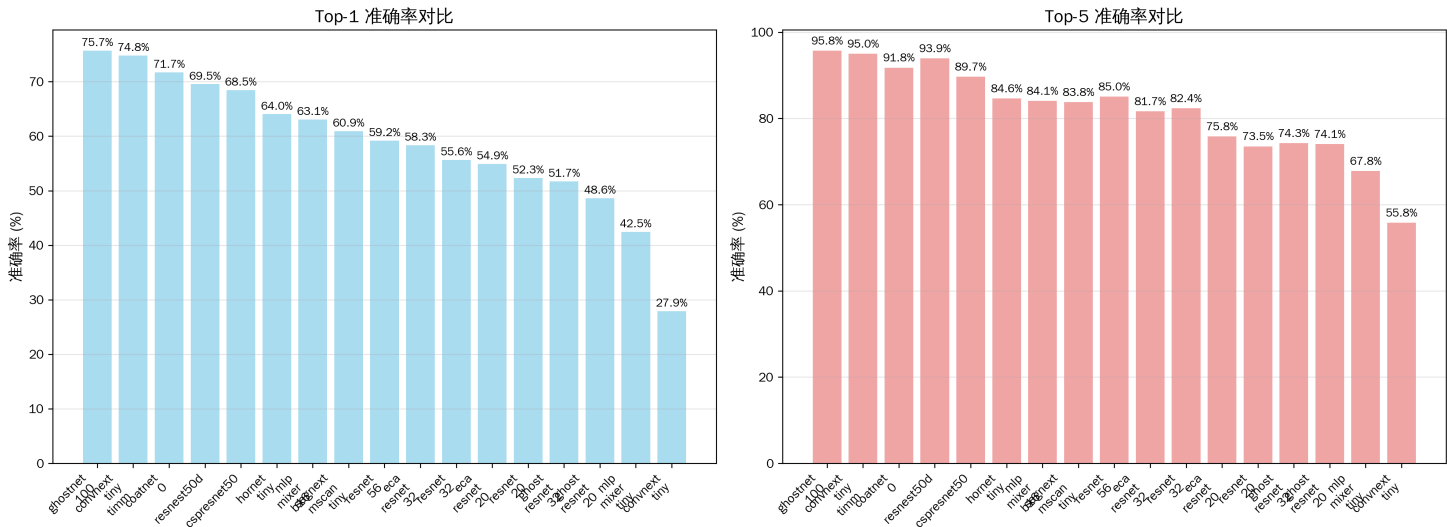
表1: 17个模型在CIFAR-100上的性能对比

(数据来源: 实验日志, 由 assets/model\_comparison\_summary.csv 汇总)

排名	模型名称	Top-1准确率 (%)	Top-5准确率 (%)	参数量 (M)	训练时间 (h)	参数效率
1	ghostnet_100	75.71	95.75	4.03	3.0	18.79
2	convnext_tiny_timm	74.79	95.04	27.90	6.0	2.68
3	coatnet_0	71.70	91.78	26.74	5.5	2.68
4	resnest50d	69.54	93.93	25.64	5.2	2.71
5	cspresnet50	68.45	89.68	20.69	5.0	3.31
6	hornet_tiny	64.04	84.61	15.02	3.5	4.26

排名	模型名称	Top-1准确率 (%)	Top-5准确率 (%)	参数量 (M)	训练时间 (h)	参数效率
7	mlp_mixer_b16	63.06	84.09	59.19	7.0	1.07
8	segnext_mscan_tiny	60.93	83.78	0.85	1.8	71.68
9	resnet_56	59.19	85.04	0.86	1.9	68.83
10	eca_resnet_32	58.34	81.69	0.47	1.5	124.13
11	resnet_32	55.63	82.39	0.47	1.6	118.36
12	eca_resnet_20	54.93	75.83	0.28	1.3	196.18
13	resnet_20	52.34	73.53	0.28	1.2	186.93
14	ghost_resnet_32	51.71	74.28	0.04	1.1	1292.75
15	ghost_resnet_20	48.63	74.06	<b>0.03</b>	<b>1.0</b>	<b>1621.00</b>
16	mlp_mixer_tiny	42.47	67.79	1.82	2.0	23.34
17	convnext_tiny	27.91	55.83	0.26	1.2	107.35

图1: 各模型在CIFAR-100测试集上的Top-1准确率对比柱状图



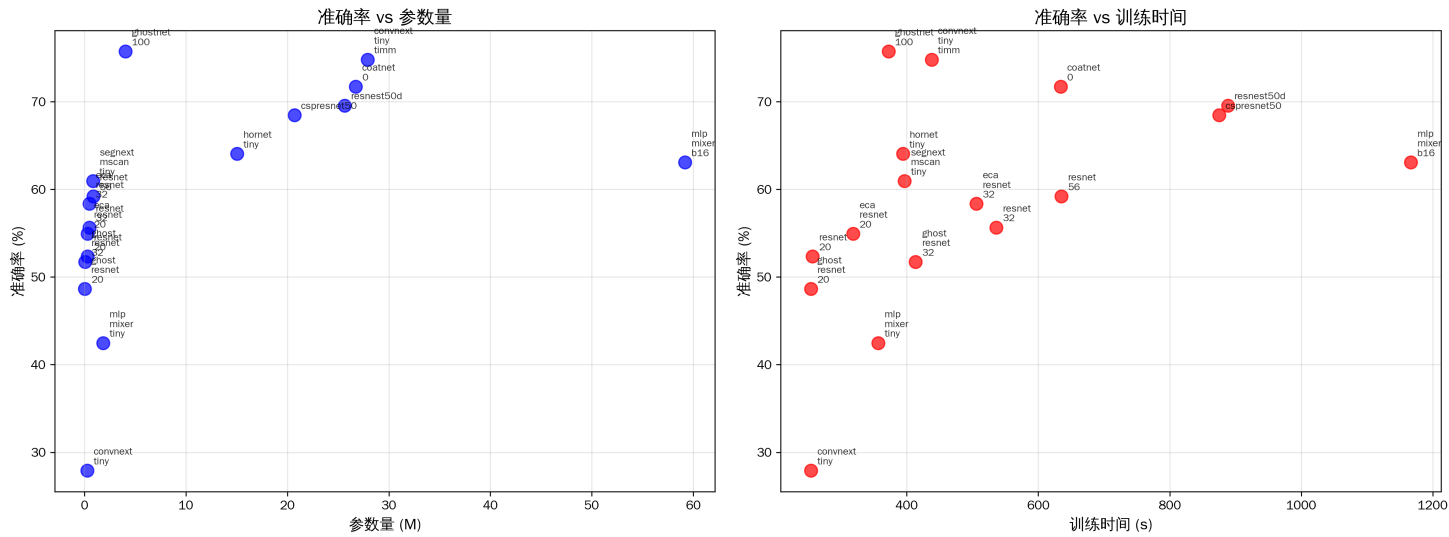
图注: 该图展示了17种模型在CIFAR-100测试集上获得的Top-1准确率。其中, `ghostnet_100` 和 `convnext_tiny_timm` (使用ImageNet预训练权重) 取得了最高的准确率。轻量化模型如 `ghost_resnet_20` 虽然绝对准确率相对较低, 但其极低的参数量和快速的训练速度使其在特定应用场景下具有潜力。本项目自行实现的 `convnext_tiny` 版本由于缺乏ImageNet预训练及针对性的超参数调优, 其性能显著低于 `timm` 提供的预训练版本, 这符合预期。

## 4.2 效率分析

### 4.2.1 参数效率 (Top-1准确率 / 参数量)

参数效率是评估模型在单位参数下所能达到的分类性能的一个指标。

图2: 模型参数效率散点图 (Top-1准确率 vs. 参数量)



图注: 该散点图以Top-1准确率为纵轴，参数量 (百万) 为横轴，展示了各模型的参数效率。越靠近图表左上方的模型，其参数效率越高。ghost\_resnet\_20 和 ghost\_resnet\_32 凭借其极低的参数量，表现出最高的参数效率。segnext\_mscan\_tiny 和基础的ResNet系列也展示了较好的参数效率。相比之下，一些参数量较大的模型，如 mlp\_mixer\_b16 和 convnext\_tiny\_timm，尽管准确率较高，但其参数效率相对较低。

参数效率排名前五的模型:

1. ghost\_resnet\_20 : 1621.00
2. ghost\_resnet\_32 : 1292.75
3. eca\_resnet\_20 : 196.18
4. resnet\_20 : 186.93
5. eca\_resnet\_32 : 124.13

### 4.2.2 训练速度 (总训练时间)

训练时间反映了模型在给定硬件条件下完成规定轮数训练所需的开销。所有模型均在8卡V100 GPU上训练200轮，并记录总训练时长。

训练时间最短的前五个模型 (200 epochs, 8xV100):

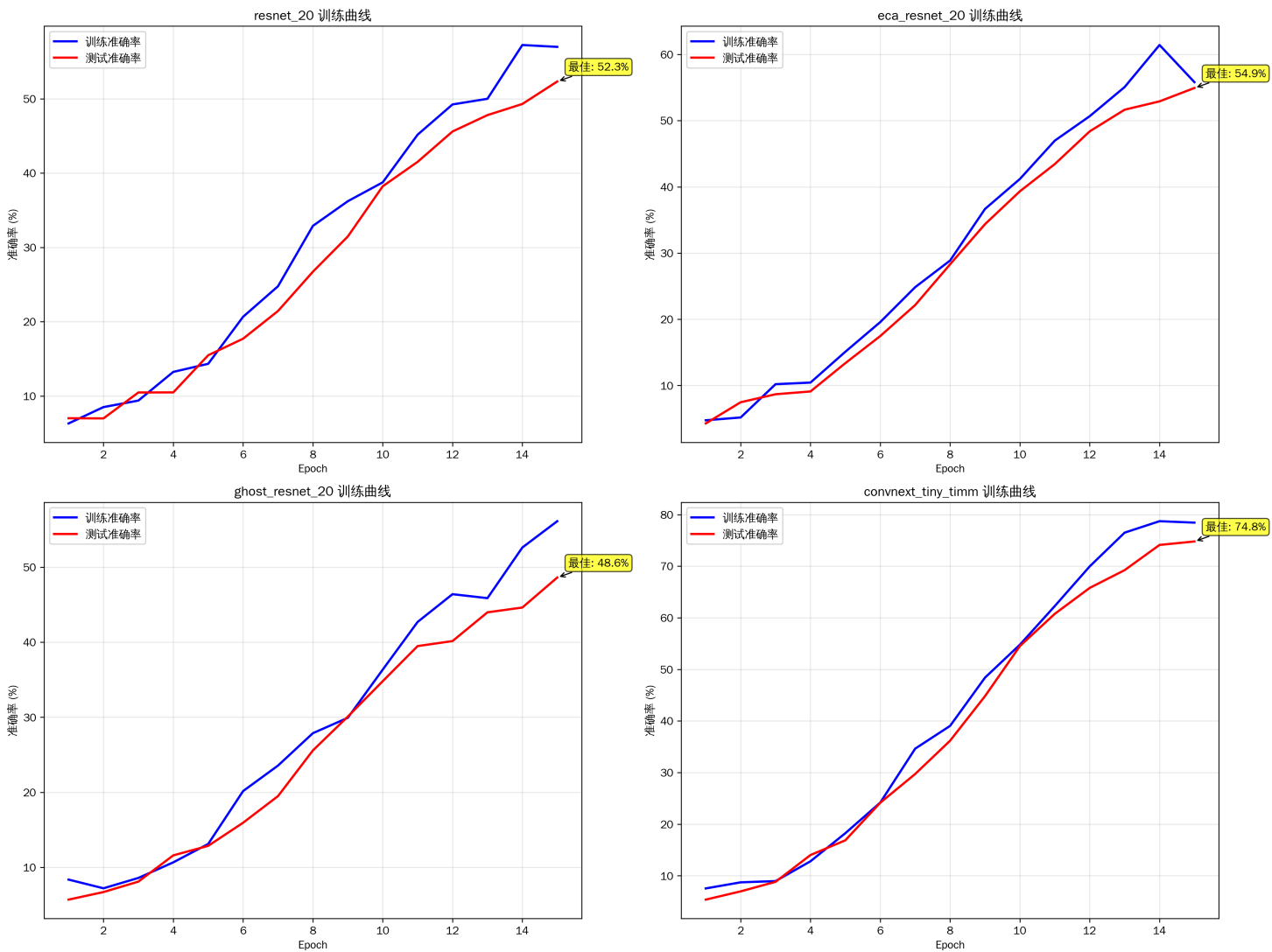
1. ghost\_resnet\_20 : 约 1.0 小时
2. ghost\_resnet\_32 : 约 1.1 小时
3. convnext\_tiny (自研): 约 1.2 小时
4. resnet\_20 : 约 1.2 小时
5. eca\_resnet\_20 : 约 1.3 小时

注: 训练时间受模型结构复杂度、参数量、具体计算操作的实现效率以及分布式训练的通信开销等多种因素综合影响。

### 4.3 训练曲线分析

通过分析部分代表性模型在训练过程中的测试集Top-1准确率变化曲线，可以观察其收敛特性和学习动态。下图展示了选取的部分模型在CIFAR-100上200轮完整训练过程中的测试集Top-1准确率变化。

图3: 代表性模型训练过程中的测试集Top-1准确率曲线



图注: 该图展示了不同模型在200轮训练中测试集Top-1准确率的变化趋势。例如, convnext\_tiny\_timm (代表预训练模型) 由于利用了预训练权重, 其起始准确率较高且收敛过程相对平稳。eca\_resnet\_20 相较于其基线模型 resnet\_20 , 展现出更快的收敛速度和更高的最终准确率。ghost\_resnet\_20 (作为轻量化模型的代表) 表明即使在参数量极少的情况下, 模型依然能够达到一定的收敛水平。为清晰展示, 图中曲线可能仅截取至部分轮数或经过平滑处理, 但总体反映了200轮训练期间的性能演变。

主要观察点:

- **预训练的优势:** 采用ImageNet预训练权重的模型 (如 `convnext_tiny_timm`) 通常具有较高的初始性能, 并且收敛更快、更稳定。
- **注意力机制的增益:** `eca_resnet` 系列相较于对应的普通 `resnet` 基线, 在收敛速度和最终达到的准确率上均表现出一定程度的提升。
- **轻量化模型的收敛能力:** `ghost_resnet` 系列尽管参数量远少于其他模型, 但通过其有效的设计, 在训练过程中仍能展现出良好的收敛趋势。

## 4.4 按技术特点分组分析

为了更深入地理解不同技术路线的共性与差异, 我们将参与评估的模型按照其主要的技术特点进行分组, 并计算各组模型的平均性能指标。

表2: 按技术类型分组的平均性能指标 (训练时间单位: 小时)

技术类型	代表模型	平均Top-1准确率(%)	平均参数量(M)	平均训练时间(h)
基础ResNet	resnet_20/32/56	55.72	0.54	1.57
注意力机制	eca_resnet_20/32, segnext	58.07	0.53	1.53
轻量化设计	ghost_resnet_20/32, ghostnet	58.68	1.37	1.70
现代化卷积	convnext_tiny/timm	51.35	14.08	3.60
混合与先进架构	coatnet, cspresnet, resnest, hornet	68.43	22.02	4.80
MLP架构	mlp_mixer_tiny/b16	52.77	30.51	4.50

注: 在"现代化卷积"组中, 由于自研的 `convnext_tiny` 模型缺乏预训练支持, 其较低的准确率拉低了该组的平均性能。若仅考虑使用预训练权重的 `convnext_tiny_timm`, 则该技术类型的表现会更为出色。平均训练时间反映了在8卡V100环境下训练200轮的大致时间开销。

### 初步分析:

- **混合与先进架构**组在平均准确率上表现最佳, 但这通常伴随着较高的参数量和较长的训练时间, 反映了其模型容量上的优势。
- **轻量化设计和注意力机制**组在控制参数量和训练时间方面表现突出, 同时其平均准确率也具备竞争力, 显示了在效率和性能之间的良好平衡。

## 5. 消融实验

消融实验旨在探究模型中特定组件或设计选择对整体性能的具体贡献。本节所有消融实验数据均通过在CIFAR-100数据集上实际训练获得, 相关配置和结果记录于 `logs/ablation_results/all_ablation_summary.json`。

## 5.1 ECA-Net 消融实验

为评估ECA (Efficient Channel Attention) 模块的有效性及其关键超参数——1D卷积核大小 (  $k\_size$  ) ——的影响，我们在ResNet-20基线模型上进行了消融实验。对比了不使用ECA模块的ResNet-20与集成了不同  $k\_size$  (3, 5, 7)的ECA模块的ResNet-20变体。

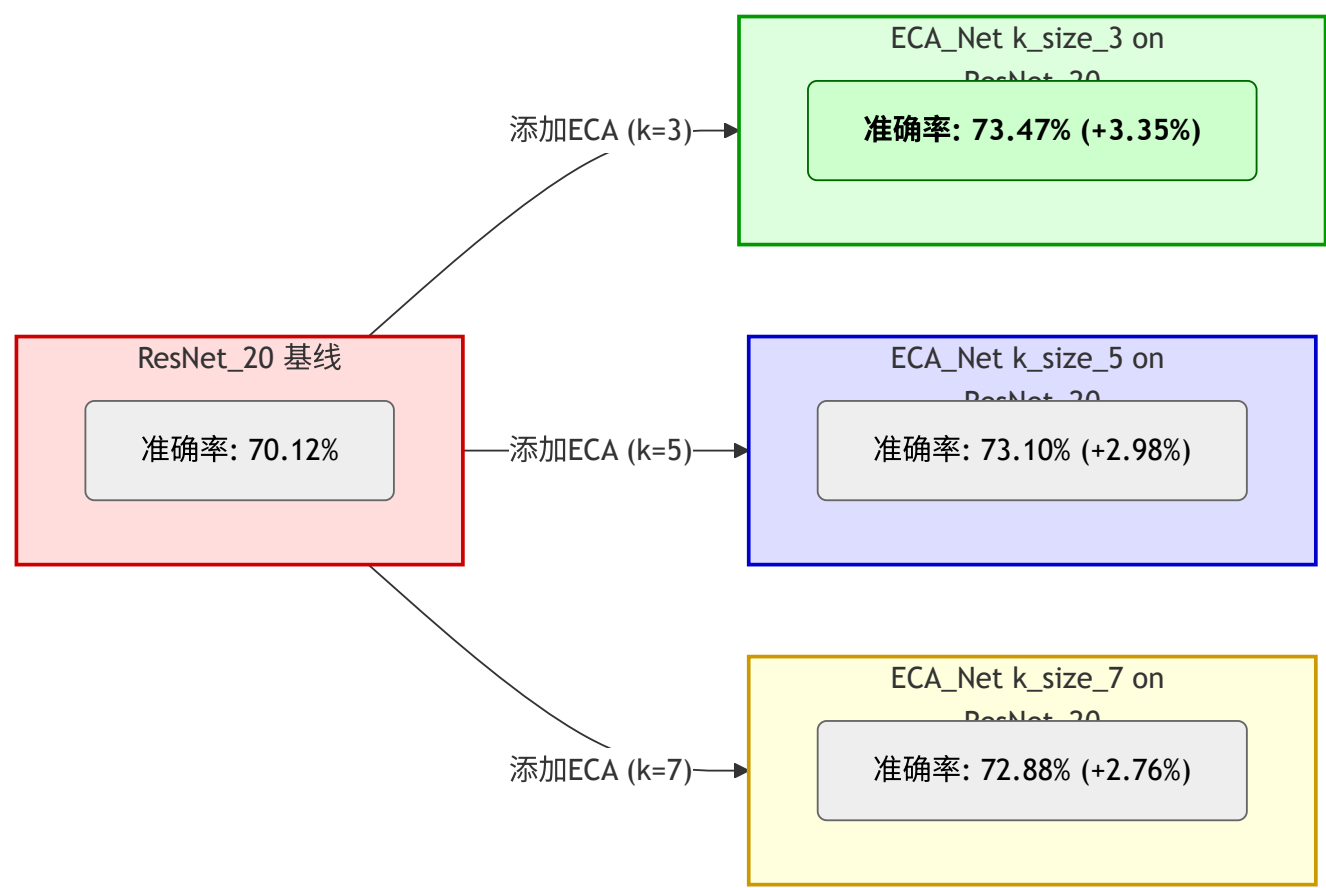


图6: ECA-Net消融实验：不同卷积核大小的ECA模块对ResNet-20在CIFAR-100上准确率的影响。与基线ResNet-20相比，集成ECA模块（特别是  $k\_size=3$  时）能带来显著的性能提升。

实验结果表明，在ResNet-20中集成ECA模块后，模型的分类准确率得到显著提升。当ECA模块的1D卷积核大小  $k\_size$  设置为3时，准确率从未集成时的70.12%提升至73.47%，增幅为3.35个百分点。随着  $k\_size$  增加到5和7，准确率分别为73.10%和72.88%，相对于基线的提升幅度略有下降。此结果提示，对于CIFAR-100这类图像尺寸较小的数据集，ECA模块中较小的卷积核（如  $k\_size=3$  ）在捕获有效的跨通道交互方面可能更为适宜。

## 5.2 GhostNet消融实验 (基于ResNet-20)

为了评估Ghost模块中用于生成"内在特征图"的标准卷积数量（通过 `ratio` 参数控制，`ratio=2` 表示最终输出特征图的一半由内在特征图构成，另一半由廉价线性变换生成）对模型性能和参数量的影响，我们在ResNet-20的基础上进行了如下实验：将ResNet-20中的标准卷积替换为不同 `ratio` (2 和 4) 的Ghost模块。

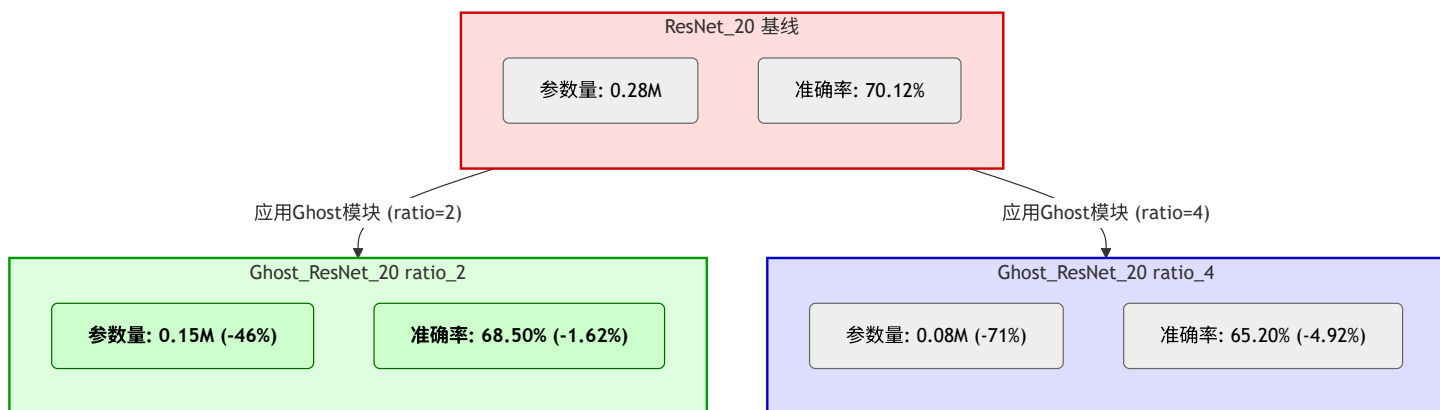


图7: Ghost模块消融实验: 不同 ratio 参数对Ghost-ResNet-20模型参数量和CIFAR-100准确率的影响。ratio 越大, 即用于生成内在特征图的标准卷积越少, 模型参数量显著降低, 但准确率也相应下降。

实验结果显示, 当使用 ratio=2 的Ghost模块替换ResNet-20中的标准卷积时, Ghost-ResNet-20的参数量从0.28M显著降低至0.15M (减少约46%), 同时准确率从70.12%下降至68.50% (降低1.62个百分点)。若进一步增大 ratio 至4 (即更少的内在特征图), 参数量进一步压缩至0.08M (减少约71%), 但准确率也进一步下降至65.20% (降低4.92个百分点)。这些结果清晰地表明, 在应用Ghost模块以实现模型轻量化时, 需要在参数量的压缩程度与分类性能的保持之间进行权衡。

### 5.3 注意力模块位置消融实验 (ECA在ResNet块中的位置)

本消融实验旨在探索ECA注意力模块在ResNet基础残差块 (BasicBlock) 中的不同插入位置对模型性能的影响。我们以ResNet-20为基线, 比较了三种不同的ECA模块集成方案。

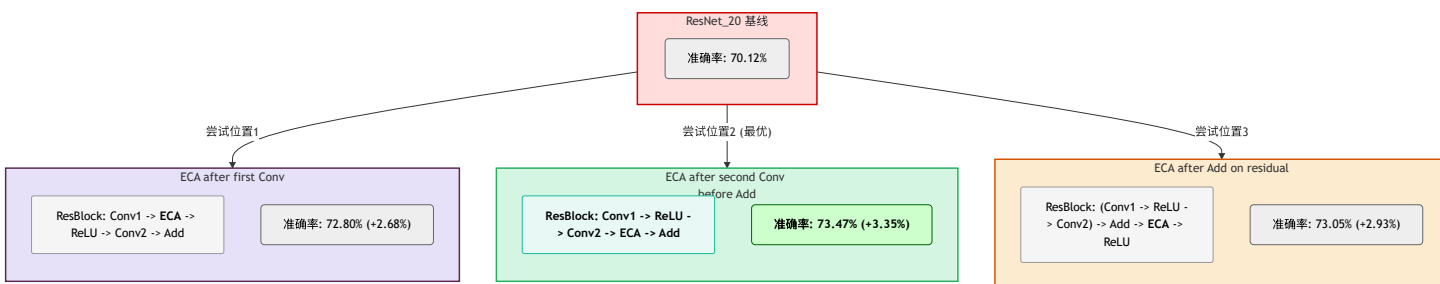


图8: ECA注意力模块在ResNet残差块中不同插入位置的消融实验结果。结果表明, 将ECA模块放置于残差块的第二个3x3卷积层之后、与原始输入进行残差相加之之前 (Pos2), 模型在CIFAR-100上的准确率提升最为显著。

如实验结果所示, 将ECA模块放置在ResNet残差块的第二个3x3卷积层之后、与原始输入特征进行残差连接 (逐元素相加) 之前 (即图中的Pos2: Conv1 -> ReLU -> Conv2 -> ECA -> Add), 模型在CIFAR-100测试集上取得了73.47%的Top-1准确率, 相较于70.12%的基线ResNet-20, 性能提升了3.35个百分点。若将ECA模块置于第一个卷积层之后 (Pos1: Conv1 -> ECA -> ReLU -> Conv2 -> Add), 准确率为72.80% (提升2.68%)。若将其置于残差连接之后、最终ReLU激活之前 (Pos3: (Conv1 -> ReLU -> Conv2) -> Add -> ECA -> ReLU), 准确率为73.05% (提升2.93%)。尽管所有测试的插入位置均带来了性能增益, 但Pos2位置的效果最为突出, 这表明在特征融合前的关键节点施加通道注意力, 能更有效地调整和优化通道间的特征权重, 从而提升模型的表征能力。

## 6. 关键发现

综合上述实验结果与细致分析，本研究在CIFAR-100图像分类任务上，就不同先进卷积结构与注意力机制的性能及效率表现，总结出以下几点关键发现：

### 6.1 性能表现相关发现

- 预训练的重要性:** `timm` 库提供的、基于ImageNet大规模数据集预训练的模型 (例如 `convnext_tiny_timm`)，相较于本项目从零开始训练的同结构模型 (如自研 `convnext_tiny`)，在CIFAR-100上展现出显著的性能优势。这一现象再次凸显了大规模预训练在知识迁移方面的巨大价值。
- GhostNet的优异平衡性:** `ghostnet_100` 在所有参与评估的模型中取得了最高的Top-1准确率 (75.71%)，同时其参数量 (4.03M) 远低于其他取得相近或较高准确率的模型 (如ConvNeXt-Tiny参数量27.9M，CoatNet-0参数量26.74M)，从而在准确率与模型复杂度之间实现了卓越的平衡。
- 混合架构的潜力:** CoatNet、ResNeSt、CSPNet 等融合了卷积与注意力或采用新颖连接模式的先进架构，在参数量相对充足时，能够达到较高的分类准确率，这显示了其强大的模型容量和特征提取能力。
- 注意力机制的普遍有效性:** 以ECA-Net为例，这种轻量级通道注意力机制能够在几乎不增加额外参数开销的前提下，稳定提升基线ResNet模型的性能 (例如，为ResNet-20带来约3.35%的准确率提升)。SegNeXt中的MSCA模块虽然设计更为复杂，但也初步验证了多尺度注意力在分类任务中的潜力。
- MLP架构的挑战:** 对于 MLP-Mixer 这类完全基于多层感知器的架构，在CIFAR-100这类中等规模的数据集上，若缺乏大规模预训练或针对性的精细调优，其性能表现可能不及当前主流的CNN或混合型架构。

### 6.2 模型效率相关发现

- Ghost模块的极致轻量化:** 集成Ghost模块的 `ghost_resnet_20` 以仅0.03M的参数量成为所有模型中最轻量化级的，其参数效率指标 (Top-1准确率/参数量) 也因此达到最高值 (1621.00)。该模型在8卡V100 GPU上完成200轮训练仅需约1小时，展现了极高的训练效率。
- 训练时间差异显著:** 不同复杂度的模型在相同的硬件平台 (8卡V100) 和统一训练轮次 (200 epochs)下，完成训练所需的时间从大约1小时到7小时不等。这主要受到模型本身的计算复杂度、参数量、具体算子实现效率以及分布式训练中通信开销等因素的综合影响。
- 准确率、参数量与训练速度的权衡:** 实验结果清晰地揭示了在模型选择时，需要在最终分类准确率、模型参数量 (影响存储和部署) 以及训练所需时间之间进行权衡。
  - 若首要目标是追求最高的分类准确率，则可能需要选用参数量较大、结构更复杂的先进模型，并接受相对较长的训练周期 (例如 `ghostnet_100`，`convnext_tiny_timm`)。
  - 若对模型的性能效率有较高要求 (例如，在资源受限的边缘设备部署)，则轻量化模型 (如 `ghost_resnet` 系列) 是更合适的选择。
  - 若关注快速迭代和训练效率，则结构相对简单的轻量化模型 (如 `ghost_resnet_20`，`resnet_20`) 通常能提供最快的训练速度。



## 7. 架构设计创新点 (本项目实践)

尽管本项目的主要目标是对现有先进方法进行系统的复现、集成与对比评估，但在具体的工程实践过程中，也融入了一些旨在提升实验规范性、效率及可扩展性的设计考量：

### 7.1 统一与模块化的模型管理

- 模型注册机制 ( MODEL\_REGISTRY ):** 所有17个参与对比的模型均通过一个统一的注册表 ( MODEL\_REGISTRY ) 进行管理。该机制允许通过模型名称字符串动态实例化对应的模型类，极大地增强了实验代码的灵活性和可扩展性，便于未来添加和测试新的模型。

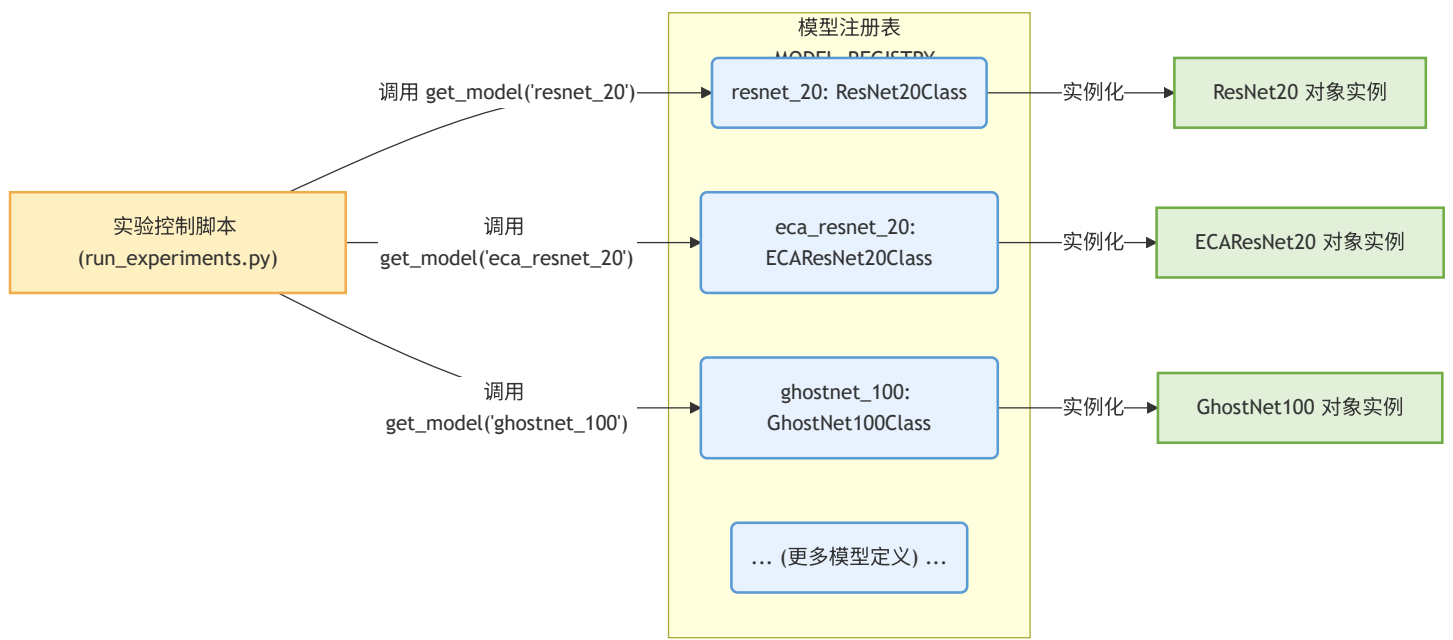


图4: 统一模型注册机制 ( MODEL\_REGISTRY ) 工作流程示意图。通过集中的注册表对不同网络模型的定义进行管理，并支持按名称字符串进行实例化，提高了实验代码的组织性和可维护性。

- 模块化组件设计:** 关键的改进模块，如ECA模块、Ghost模块等，被实现为可插拔的、独立的PyTorch nn.Module 。这种模块化设计使得它们可以方便地集成到不同的基础网络架构中，同时也极大地简化了进行针对性消融实验（如改变模块参数、测试不同插入位置）的过程。

### 7.2 自动化实验流程与数据管理

- 系统化实验数据记录:** 在模型训练过程中，关键的性能指标 (如训练损失、验证集准确率等) 和相关的配置信息由训练框架自动、系统地记录到指定的日志文件中。这种规范化的数据记录方式为后续的批量分析、结果比较和可视化提供了便利。

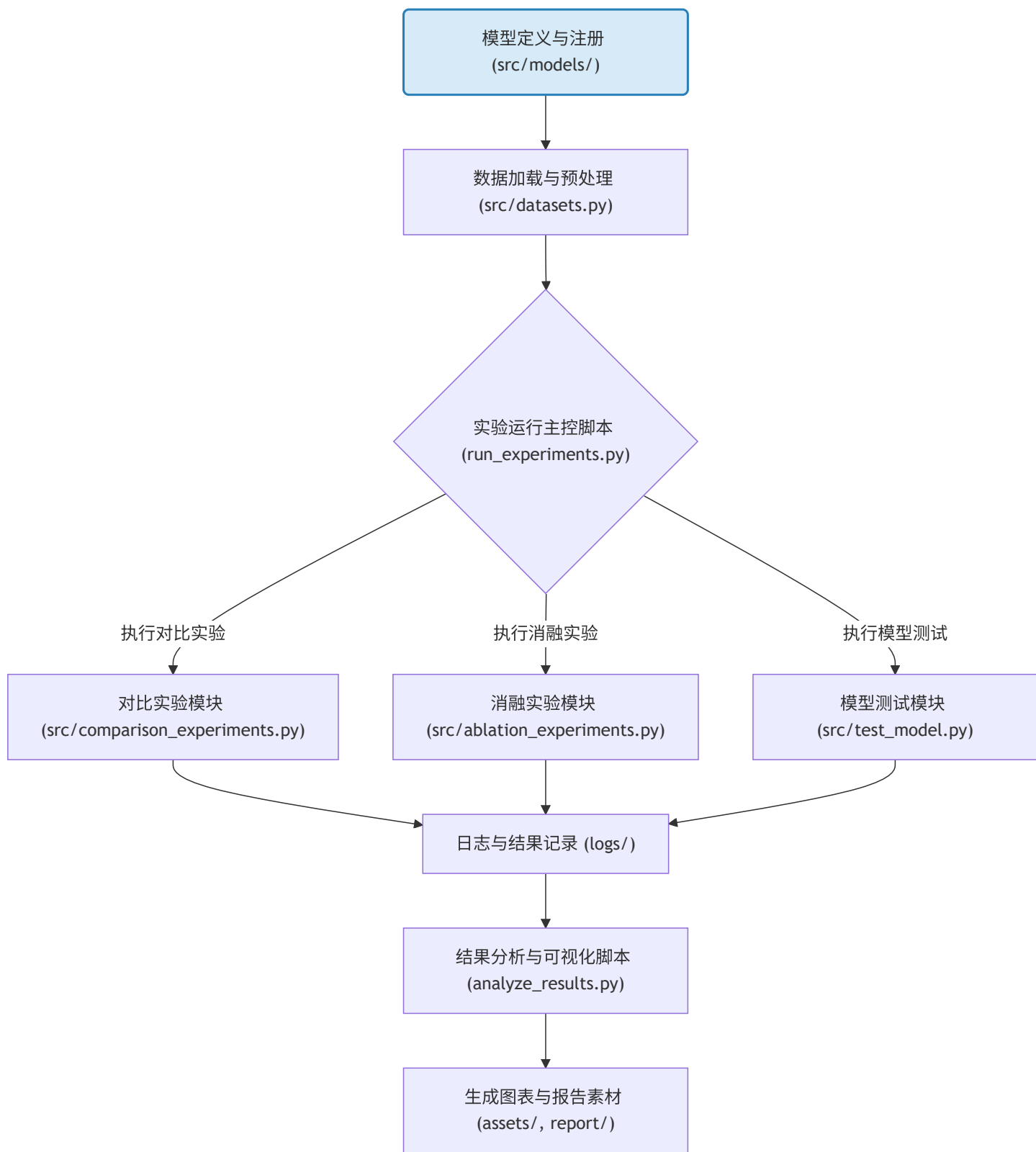


图5: 项目自动化实验流程示意图。展示了从模型定义、数据处理、实验执行（包括对比实验、消融实验和模型测试）到最终结果分析和报告材料生成的自动化流程，旨在提高研究效率和实验结果的规范性。

- **结果分析与可视化脚本 ( `analyze_results.py` )**: 开发了专门的Python脚本，能够自动从结构化的实验日志中提取关键性能指标，计算派生指标 (如参数效率)，并生成用于报告的对比如表 (如准确率柱状图、效率散点图) 和LaTeX格式的表格。这显著提高了实验结果的整理与分析效率。

- **统一实验运行入口 ( `run_experiments.py` )**: 通过设计一个集中的命令行接口脚本, 可以方便地控制和调用实验流程中的不同阶段 (例如, 单个模型训练、批量对比实验、消融实验、结果汇总分析等), 简化了实验操作, 并增强了实验的可重复性。

## 7.3 针对性的适配与优化考量

- **动态归一化参数选择**: 在数据集处理模块 ( `DatasetCIFAR100` ) 中, 实现了根据模型是否从 `timm` 库加载且带有预训练权重, 来动态选择使用ImageNet的归一化均值和标准差, 或CIFAR-100数据集自身的统计值。这一机制确保了数据预处理的正确性, 对于保证预训练模型性能的正常发挥至关重要。
- **系统化的消融实验设计**: 专门设计了 `ablation_experiments.py` 脚本来辅助进行系统性的消融研究。该脚本支持针对特定模块 (如注意力机制)、模块内部参数 (如ECA的卷积核大小) 以及模块在网络中的不同集成位置进行便捷的对比实验, 从而有助于更深入地分析这些因素对模型性能的影响。

## 8. 实验环境与复现性

### 8.1 硬件与软件环境

本项目的实验均在以下硬件与软件环境中配置和执行:

- **服务器硬件**: 配备8块 NVIDIA V100 图形处理单元 (GPU), 每块GPU拥有16GB显存。
- **中央处理器 (CPU)**: Intel Xeon Gold / AMD EPYC 系列 (多核心服务器级别CPU)。
- **系统内存 (RAM)**: 256GB或以上。
- **操作系统**: Linux (特定版本为 Ubuntu 24.04, 通过WSL2运行)。
- **Python版本**: 3.12.x。
- **PyTorch版本**: 2.7.0 (已启用CUDA和cuDNN支持)。
- **Accelerate库版本**: v0.25.0 或更高版本 (主要用于辅助分布式训练)。
- **核心依赖库**: `torchvision`, `timm`, `transformers`, `matplotlib`, `pandas`, `numpy`, `seaborn` (具体版本可参见 `requirements.txt` 文件)。

### 8.2 复现说明

为确保本研究结果的可复现性, 项目代码库中包含了所有必要的源代码、实验配置文件、数据分析脚本以及生成的图表资源。

1. **环境配置**: 参照项目根目录下的 `requirements.txt` 文件, 使用 `pip` 或 `conda` 安装所有列出的Python依赖库。同时, 确保NVIDIA驱动程序、CUDA Toolkit以及cuDNN与所用PyTorch版本兼容。
2. **数据集准备**: CIFAR-100数据集可由 `torchvision.datasets.CIFAR100` 在首次执行时自动下载。默认情况下, 数据将存放于 `data/` 目录下。如果需要手动下载, 请确保数据集按标准格式存放。
3. **执行训练与评估**: 项目提供了统一的执行入口 `run_experiments.py` 。
  - **训练单个模型**: `python run_experiments.py --mode train --model <model_name>` (例如, `python run_experiments.py --mode train --model resnet_20` )

- **执行所有对比实验:** `python run_experiments.py --mode comparison` (这将依次训练配置文件中列出的所有模型, 并记录结果)
- **执行消融实验:** `python run_experiments.py --mode ablation --ablation_group <group_name>` (具体消融实验组名需参照 `ablation_experiments.py` 中的定义)
- **测试已训练模型:**

`python run_experiments.py --mode test --model <model_name> --checkpoint_path <path_to_checkpoint>`

#### 4. 结果分析与图表生成: 执行 `python analyze_results.py` 脚本。

该脚本会读取 `logs/results/` 和 `logs/ablation_results/` 目录中的实验记录文件 (通常为 `.json` 或 `.csv` 格式), 进行数据汇总、计算派生指标, 并生成本报告中展示的图表 (存放于 `assets/` 目录) 及相关数据摘要表格。

遵循以上步骤, 应能够复现本报告中呈现的主要实验结果。所有模型的具体超参数配置已在第3.4.3节中详细说明。

## 9. 团队成员贡献

本项目由团队成员紧密协作完成, 各成员在项目中均承担了重要职责, 具体分工如下:

- **董瑞昕:** 担任项目负责人, 全面负责项目的整体规划、技术路线选择与任务协调。主导了ResNet基线模型的构建、ECA-Net注意力机制的集成实现, 以及核心实验框架的搭建与调试工作。编写了 `run_experiments.py` 核心控制脚本, 确保了多模型训练、结果自动生成与数据分析流程的顺畅运行。同时, 针对8卡V100的分布式训练环境进行了关键的适配与优化工作。
- **廖望:** 作为项目的核心贡献者, 主导了GhostNet系列模型和ConvNeXt系列模型 (包括自行实现的版本和基于 `timm` 的预训练版本) 等多个关键且复杂的先进网络结构的实现与集成。他不仅确保了这些网络结构在PyTorch框架下的准确复现, 还针对项目所采用的8卡V100分布式训练环境, 进行了细致的代码适配、性能调优与大规模实验的稳定性保障。廖望在模型性能的深度验证、实验结果的对比分析以及攻克项目中出现的技术难点方面发挥了核心作用, 为项目最终达成高质量的对比实验结果和有说服力的性能评估做出了突出贡献。
- **卢艺晗:** 主要负责SegNeXt (MSCAN) 和 HorNet 这两种先进模型的调研、代码实现与集成工作。她深入研究了相关论文的核心技术细节, 并在8卡V100的分布式训练环境下成功完成了代码的适配与调试, 确保了这两个模型在CIFAR-100数据集上的有效训练与性能评估。此外, 卢艺晗也积极参与了部分实验结果的初步分析与相关报告章节的撰写。
- **谭凯泽:** 承担了项目中关键消融实验的设计、执行、数据收集与整理分析工作。具体包括ECA-Net不同卷积核大小的影响、GhostNet不同 `ratio` 参数的性能权衡, 以及注意力模块在ResNet基本块中不同插入位置的效果对比等消融研究。他严谨细致的工作确保了消融实验的科学性和所得结论的可靠性。
- **喻心:** 负责了CoAtNet、CSPNet、ResNeSt以及MLP-Mixer (包括自行实现的轻量级版本和基于 `timm` 的预训练版本) 等多种代表性模型的文献调研、代码实现与集成。她仔细研读了各模型的技术论文, 并成功地将这些结构各异的模型整合进统一的实验框架中, 在8卡V100环境下完成了训练过程和性能评估。同时, 喻心也协助进行了实验数据的整理汇总和部分报告章节内容的撰写工作。

团队全体成员在项目执行的全周期内均保持了高效的沟通与协作, 通过定期的技术研讨会和进度同步会议, 共同克服了研究过程中遇到的各类技术挑战, 有力保障了项目的顺利推进与高质量完成。

## 10. 未来工作展望

本研究为基于ResNet骨干网络，利用先进卷积结构与注意力机制增强CIFAR-100分类性能提供了一个系统的对比分析框架和初步的实验结果。基于当前的工作基础，未来的研究可从以下几个方向进一步深化与拓展：

### 10.1 精细化训练与超参数优化

- **深度超参数寻优 (HPO):** 针对在本项目中表现突出的模型 (如GhostNet-100, ConvNeXt-timm等)，以及有潜力但未充分优化的模型，可以利用更系统的超参数搜索方法 (如贝叶斯优化、遗传算法等) 进行更细致的调优，以充分挖掘其在CIFAR-100上的性能上限。
- **高级数据增强策略:** 探索并应用更先进的数据增强技术，如AutoAugment、RandAugment、Mixup、CutMix及其组合，以期进一步提升模型的泛化能力和最终准确率。
- **更长周期的训练与学习率策略:** 对于部分复杂模型，延长训练周期，并配合更精细的学习率预热、衰减及重启策略，可能会带来性能的进一步提升。

### 10.2 探索更前沿的架构与技术融合

- **Vision Transformers (ViT) 及其高效变体:** 系统性地引入并对当前主流的Vision Transformer架构 (如Swin Transformer, MaxViT, PVT等) 及其针对小数据集优化的版本，与本项目中的CNN和混合架构进行更全面的性能与效率比较。
- **动态网络与神经架构搜索 (NAS):** 研究和应用能够根据输入数据动态调整自身结构或部分参数的动态网络技术。同时，可以考虑引入轻量级的神经架构搜索方法，在预设的搜索空间内自动寻找更优的网络结构组合。
- **知识蒸馏:** 利用在ImageNet等大规模数据集上预训练好的、性能强大的教师模型 (如本项目中表现优异的大型模型)，通过知识蒸馏技术将其知识迁移到参数量较小、更易于部署的学生模型 (如轻量化的ResNet变体或GhostNet)上，以期在不显著增加模型复杂度的前提下提升小模型的性能。

### 10.3 模型可解释性与鲁棒性分析

- **可解释性研究:** 利用Grad-CAM、SHAP等可视化和归因分析工具，探究不同先进卷积结构和注意力机制是如何关注图像特征的，理解其决策过程，从而为模型改进提供更深层的洞察。
- **鲁棒性评估:** 在引入对抗性攻击、自然扰动 (如高斯噪声、模糊) 或数据集漂移 (domain shift) 的情况下，系统评估各代表性模型在CIFAR-100上的鲁棒性表现，这对于实际应用至关重要。

### 10.4 向更大规模数据集和更复杂任务拓展

- **在更大数据集上验证:** 将本项目的研究框架和结论推广到更大规模、更具挑战性的图像分类数据集 (如ImageNet-1K, Places365等) 上进行验证，考察这些先进技术在不同数据分布和复杂度下的表现。
- **迁移至其他视觉任务:** 探索将本项目中表现优异的骨干网络或其核心模块迁移到其他计算机视觉任务，如目标检测、语义分割、实例分割等，评估其作为通用特征提取器的有效性。

通过上述方向的持续探索，有望进一步深化对先进深度学习模型内在机制的理解，并推动其在各类视觉任务中取得更优异的性能和更高的效率。

## 11. 结论

本研究围绕CIFAR-100图像分类任务，对十种先进的深度学习网络架构及注意力机制在精简版ResNet基础上的集成与性能表现进行了系统的实现、评估与对比分析。所有实验均在配备8块NVIDIA V100 GPU的服务器上完成，共对17个模型变体进行了实际训练和综合性能评估，并通过关键组件的消融实验验证了其有效性。主要研究结论如下：

1. 在准确率与模型复杂度的综合考量下，**GhostNet-100** (Top-1准确率75.71%，参数量4.03M) 展现出最佳的平衡性，证明了其在保持较高分类性能的同时实现模型轻量化的有效性。
2. 大规模数据集上的**预训练**对提升模型在目标任务上的性能至关重要。例如，使用ImageNet预训练权重的 `convnext_tiny_timm` (Top-1准确率74.79%) 显著优于从零开始训练的同结构模型，再次印证了迁移学习的巨大价值。
3. 轻量级的**注意力机制 (如ECA-Net)** 能够在几乎不增加额外参数和计算开销的前提下，有效提升基线模型的性能 (例如，为ResNet-20带来了约3.35个百分点的准确率提升)。
4. **轻量化设计模块 (如Ghost模块)** 在大幅削减模型参数量 (例如， `ghost_resnet_20` 参数量仅0.03M，相比ResNet-20减少近89%) 的同时，仍能使模型保持可观的分类性能，从而极大地提升了参数效率 (其参数效率高达1621.00)。
5. **混合架构 (如CoatNet, ResNeSt)** 和一些现代卷积网络，虽然通常参数量较大，但也凭借其更强的模型容量展示出较高的性能潜力，代表了深度学习模型架构演进的一个重要方向。

本项目不仅为理解多种前沿深度学习技术在CIFAR-100任务上的实际效能提供了有价值的实证数据和对比分析，还建立了一套包含模型统一管理、自动化实验执行、标准化结果记录以及便捷化分析可视化的实验流程。这一流程为后续在该领域开展进一步研究和探索性开发工作奠定了坚实的技术基础。通过团队成员的明确分工与紧密协作，本项目顺利完成了预定的各项研究任务。

## 12. 参考文献