杭州电子科技大学

硕 士 学 位 论 文

题　　目：基于深度学习的 H.265/HEVC 编码技术研究

研 究 生　殷先英

专　　业　电子信息

指导教师　周志刚　教授

完成日期　2023 年 5 月

杭州电子科技大学硕士学位论文

基于深度学习的H.265/HEVC编码技术研究

研 究 生： 殷先英

指导教师： 周志刚 教授

2023年5月

Dissertation Submitted to Hangzhou Dianzi University
for the Degree of Master

Research on H.265/HEVC Coding Technology Based on Deep Learning

Candidate: Yin Xianying

Supervisor: Prof. Zhou Zhigang

May, 2023

摘要

随着 5G 蜂窝网络、Wi-Fi 6 局域网等高速信息的部署和运营，高清视频直播和短视频应用日益成为网络业务主流，对音视频的清晰度、编解码效率提出了更高要求，支持 8K 分辨率的 H.265/HEVC 视频编解码将成为主流。H.265/HEVC 在相同的图像质量下，与 H.264/AVC 相比码流可以减少 50％左右。由于引入特色编码技术，提升了编码性能，同时也带来编码复杂度变大的问题。深度学习技术和神经网络技术的飞速发展，为降低编码复杂度，提升编解码效率指引了新的方向。论文基于深度学习技术，对 H.265/HEVC 帧内编码单元划分技术进行研究。全文的主要研究内容如下：

针对基于深度学习帧内编码单元划分算法中，神经网络层数浅、训练集数据少的问题，提出一种 CuprNet 网络编码单元划分算法，基于 ResNet18 网络设计了 CuprNet 网络，优化 ResNet18 网络全连接层输出与编码单元最小划分尺寸相匹配。构建了包含视频亮度信息和编码树单元划分结构信息的预处理数据集。在网络训练中还采用编码树单元整体亮度信息，来计算最小尺寸编码单元的深度信息，避免传统的率失真代价函数计算，加速了编码单元划分。评估结果表明，与官方编码器相比，提出的算法编码时间降低了 79.83%，码率损失为 7.943%。

针对编码树单元特征未有效利用的问题，基于分类模型提出编码单元划分分类网络 CupcNet，并提出了一种 CupcNet 网络编码单元划分算法。搭建了三层神经网络，构建了包含视频亮度信息和编码树单元划分标签的预处理数据集。在 CupcNet 网络中使用小卷积核，增强了网络非线性表达能力，减少了参数量。评估结果表明，提出的算法支持不同量化参数，与官方编码器相比，编码时间降低了 64.01%，码率损失为 2.9493%。

关键词：深度学习，视频编解码，帧内编码单元划分，神经网络

Abstract

With the deployment and operation of high-speed information networks such as 5G cellular networks and Wi-Fi 6 local area networks, high-definition video live broadcast and short video applications have increasingly become the mainstream of network services, which put forward higher requirements for audio and video clarity and encoding and decoding efficiency. 8K resolution H.265/HEVC video codec will become the mainstream. Under the same image quality, H.265/HEVC can reduce the code stream by about 50% compared with H.264/AVC. Due to the introduction of characteristic encoding technology, the encoding performance is improved, but it also brings about the problem of increased encoding complexity. The rapid development of deep learning technology and neural network technology has guided a new direction for reducing coding complexity and improving coding and decoding efficiency. Based on deep learning technology, this thesis studies the H.265/HEVC intra coding unit division technology. The main research content of the full text is as follows:

Aiming at the problem of shallow neural network layers and less training data in the intra-frame coding unit division algorithm based on deep learning, a CuprNet network coding unit division algorithm is proposed. The CuprNet network is designed based on the ResNet18 network and optimize the ResNet18 network to match the output of the fully connected layer with the minimum partition size of the encoding unit. A preprocessed data set including video brightness information and coding tree unit partition structure information is constructed. In the network training, the overall brightness information of the coding tree unit is also used to calculate the depth information of the smallest size coding unit, which avoids the traditional calculation of the rate-distortion cost function and accelerates the division of coding units. The evaluation results show that compared with the official encoder, the proposed algorithm reduces encoding time by 79.83% and has a bit rate loss of 7.943%.

Aiming at the problem that the coding tree unit features is not effectively utilized, based on the classification model, a coding unit division classification network CupcNet is proposed, and a CupcNet network coding unit division algorithm is proposed. A three-layer neural network was constructed, and a preprocessing dataset containing video brightness information and encoding tree unit partitioning labels was constructed. The small convolution kernel is used in CupcNet network to enhance the nonlinear expression ability of the network and reduce the number of parameters. The evaluation results show that the proposed algorithm supports

different quantization parameters, and compared with the official encoder, the encoding time is reduced by 64.01%, with a bit rate loss of 2.9493%.

Keywords: Deep learning, Video coding and decoding, Intra coding unit division, Neural network

目　录

第1章 绪论

1.1 研究背景与意义

视频是人们获取外界信息的重要途径，与文字、图片相比，能够更加直观地从视频中获取信息。纵观视频发展史，从最初的黑白电视至彩色电视，再到近些年随着互联网的浪潮获得迅速发展的数字视频，人们对视频清晰度有了更高的需求。在这个数字化和信息化时代，电商直播、MCN 异军突起，短视频用户规模巨大，视频在人们的日常生活和工作中无处不在，视频流量也逐年增加。5G 时代的来临，人工智能、远程监控、医学成像和视频会议等技术的发展，对互联网传输带宽和存储的要求更加严格，以满足用户视讯数据传输、存储和显示的需求。

视频是数据量非常大的信息载体，很难将原始视频在网络上进行传输，人们在生活和工作中接触到的视频，都是经过数据压缩和编码的。国际组织于 20 世纪 80 年代开始制定国际视频编解码标准，以提高视频编解码的通用性和规范性。根据国际视频编解码标准，所有编码后的码流必须遵循一定的语法结构，以确保解码器能够正常运行。随着技术的不断发展，国际组织对视频编解码标准进行不断进行更新。2013 年，国际电信联盟 ITU-T（Telecommunication Standardization Sector）发布了一项全球性的视频编解码标准——H.265/HEVC，它是一种先进的、高效的视频编解码标准。H.265 又称 HEVC（High Efficiency Video Coding），可以实现 3840×2160 的高清视频压缩，并且帧率可以达到每秒三十帧以上，为用户提供更加优质的视觉体验。与上一代国际标准 H.264/AVC 相比，H.265/HEVC 编码效率更高，视频质量更优，网络适应性也更强，因而在远程监控、高清电视、医学成像等领域获得了广泛的应用。H.265/HEVC 采用了包括环路滤波、熵编码、量化、帧间预测、变换和帧内预测等模块的混合编码框架，并在每个模块提出独特的编码技术，如利用四叉树划分编码单元，35 种不同的帧内预测方式，帧间运动信息融合，像素自适应补偿等技术。特色编码技术的引入，提升了编码性能，同时也带来编码复杂度变大的问题。

近些年来，深度学习的兴起，为 H.265/HEVC 算法优化指明了新的方向。深度学习通过构建神经网络，对数据进行多层处理并学习数据特征，广泛应用在目标识别、音频识别及计算机视觉等领域，并在这些领域取得良好的效果。近些年来，将深度学习赋能视频编解码是视频编解码领域的研究热点。与传统的视频压缩和编码算法不同，利用深度学习的编码算法充分发挥了神经网络非线性表达特性以及大数据的驱动作用，为提高视频编解码性能带来崭新的思路。因此，将视频编解码技术与深度学习结合十分有意义，它不仅是当前视频编解码领域的热门课题，又可以为新一代数字视频编解码提供技术支

持。

考虑到 H.265/HEVC 提出的利用四叉树划分编码单元 CU（Coding Unit）的技术，在官方参考软件 HM 中占用超过 80%的总编码时间，是编码复杂度提高的重要原因[1]。本文对 H.265/HEVC 帧内编码单元划分技术进行研究，应用深度学习技术，对 H.265/HEVC 帧内编码单元划分算法进一步优化，从而大大降低编码的复杂度。

1.2 国内外研究现状

1.2.1 视频编解码发展史

国际组织在 20 世纪 80 年代初期开始制定视频编解码标准，图 1.1 是视频编解码的完整流程。迄今为止，相关组织已制订多种视频编解码标准。国际标准组织 ISO（International Organization for Standards）的 MPEG（Moving Pictures Experts Group）制定了 MPEG-1、MPEG-4；国际电信联盟-电信标准部 ITU-T 制定了 H.261、H.263、H.263+、H.263++；ITU-T/ISO 联合制定了 H.262/MPEG-2、H.264、H.265 等。此外，2002 年 6 月，中国数字视频编解码组织发起了 AVS（Audio Video coding Standard）系列，为中国独立自主版权的多媒体信源标准的发展提供了强有力的支撑。

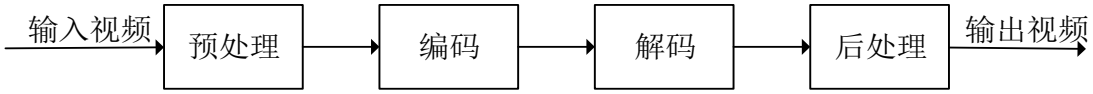


图 1.1 视频编解码流程

MPEG-X 系列是一组适用于数字运动图像的压缩标准。最早的 MPEG-1[2]制定于 1992 年，主要目的是在数字媒体上对视频进行压缩编码，但其编码能力有限，运动大的视频编码后质量差，适用范围小。针对 MPEG-1 存在的问题，MPEG 制订和推出了 MPEG-2[3]，MPEG-2 标准支持的图像格式更多，编码后的图像质量更高，主要应用于卫星电视和数字视频广播技术中。国际标准组织在 1998 年发布了 MPEG-4。MPEG-4 采用了全新的编码理念，即面向对象编码，针对对象特点，采用对应的编码方法和表示方法，主要应用于无线通信、电视电话和音频广播领域。

H.26X 是针对视频编码的标准。1990 年，ITU-T 发布了数字视频编码标准 H.261[4]。H.261 搭建了包括熵编码、变换、量化和帧间预测等模块的混合编码框架，为后续编解码标准奠定了基础。H.261 主要应用于基于综合业务数字网的视频通信。针对低码率的视频会议，国际组织推出了 H.263[5]，提高了抗误码能力和视频编解码效率。H.264/AVC[6]沿用了 H.261 的编码框架，但在每个模块采用新的技术，并使用了分层处理的算法结构，获得了远远超过以往标准的编码性能，适用于视频会议和视频流应用。

2013 年公布的 H.265/HEVC[7]编解码标准打破了 H.264/AVC 长时间处于主导地位的局面。与 H.264/AVC 标准相比，它在相同的图像质量下码流可以减少 50％左右。H.265/HEVC 通过引入四叉树分割、运动信息融合、先进矢量预测和波前并行处理技术，

提升编码性能。H.265/HEVC 编码性能的提升，伴随着其计算复杂度的提高。计算复杂度的提高主要体现在以下三点：

1．H.265/HEVC 的帧内预测模型数量大幅增加，共有 35 种预测模型，包括 DC 预测、33 种角度预测和 Planar 预测。这些模型的数量远远大于 H.264/AVC 的 17 种，使得模式选择变得更加复杂；

2．H.265/HEVC 的区域划分方式更多样化，采用四叉树划分和非对称划分技术，编码单元尺寸更多，运动补偿更加复杂；

3．H.265/HEVC 中增加了变换单元的概念，最大变换单元的大小由上一代视频编解码标准 H.264/AVC 中的 8×8 增加到现在的 32×32，运算量大大增加。

2020 年，德国服劳恩霍夫通信技术研究所（Fraunhofer HHI）公布了全新的视频编解码标准——简称 H.266[8]的通用视频编码 VVC。相对于 H.265/HEVC，新一代国际视频编解码标准 H.266/VVC 在相同图片质量下压缩比例提高 50%。H.266/VVC 沿用了 H.265/HEVC 编解码模块，但 H.266/VVC 在每个模块上都有相应的提高。H.266/VVC 采用了由四叉树，二叉树和三叉树组成的多类型树划分结构，这种划分结构与上一代视频编解码标准相比更加灵活，编码性能得到了显著改善。此外，H.266/VVC 保留了上一代标准中已有的去方块滤波和像素自适应补偿两类滤波器，同时引入特色的自适应环路滤波与亮度映射色度缩放滤波，这些滤波器可以更好地捕捉图像中的细节，从而进一步提高编码性能。

AVS 是针对国内音视频产业需求，我国自主制定的音视频信源标准，包括块划分、帧内预测、帧间预测和变换等技术。迄今为止，我国发布了 AVS[9]、AVS2[10]和 AVS3[11]三代标准编码。我国视频编解码标准虽然起步较晚，但是发展迅速，成果显著，AVS 编码性能可媲美目前广泛应用的视频编解码标准，AVS 为我国音视频发展提供了有力保障，并在国内实时通信、网络电视和视频监控等领域得到广泛应用。

1.2.2 编码单元划分研究现状

尽管现有的传统压缩和编码技术能够在某种程度上提高视频编码性能，并有效减轻视频的储存与传送的负担，然而，随着视频数据规模的不断扩大，传统的视频编码技术很难实现更高的视频编码效率。利用深度学习优化视频编解码算法是目前视频编解码标准领域研究的热点。下面对本文涉及的帧内编码单元快速划分算法进行介绍。

目前，帧内编码单元快速划分算法主要有两种，一种是基于编码特征的启发式算法，另一种是将机器学习赋能帧内编码单元划分的算法。传统的帧内编码单元快速划分算法是基于编码特征，如率失真代价、图像纹理复杂度等，或者根据统计特性人为地制定决策规则，提前终止或决定编码单元划分。早先，许多学者对传统的基于启发式的编码单元快速划分算法进行了研究。例如在文献[12]，Shen 等人利用已编码的编码单元与当前被编码的编码树单元在时空上的联系，设计出一个有效的编码单元大小决策算法。此外，

还应用了率失真优化技术来进一步减少运动估计的计算。文献[13]提出了一种具有全局复杂因子的 H.265/HEVC 编码模式选择算法。通过调节复杂度因子，在运算复杂度和压缩效率上取得了平衡。文献[14]提出基于全零块检测的算法来进行提前划分决策，通过快速的零块检测，提前终止编码单元编码，加快编码过程，一旦满足早期终止条件，它将跳过剩余的编码单元的编码。文献[15]提出了一种包括快速编码单元深度选择和对帧间 2N×2N 模式两个方面的快速决策。文献[16]研究了最优停止理论，设计了一种基于三级最优停止理论的决策方法，在早期确定每个编码树单元的最佳编码单元深度、预测单元模式和变换单元深度，减少了最佳编码参数选择的计算工作量。基于启发式的编码单元快速划分算法的有关特征需要手动提取，算法适用范围小，算法节省时间有限，对于实际应用场景中具有不同时空特性的视频序列，单一或不全面的率失真运算跳过机制会导致算法的移植性较差。

随后，机器学习被应用于视频编解码标准中以加速帧内编码单元划分，早期使用的方法主要是通过数据挖掘提取不同视频序列的特征。但该类方法过于依赖人工提取的数据，其数据本身的特征表达能力较差。近年来，随着机器学习技术的不断完善与发展，针对帧内编码单元的快速划分问题，提出了新的解决方案，以提高其分类效率和准确性。早期，基于机器学习的帧内编码单元快速划分算法将帧内编码单元划分问题视作二分类问题，力图从中抽象出规律。文献[17]提出了一种支持向量机 SVM 离线训练的 H.265/HEVC 视频编解码标准编码单元快速划分决策。文献[18]提出一种基于二进制和多类支持向量机的快速 H.265/HEVC 编码算法，该算法通过二值分类器来预测编码单元划分标志，在预测单元模块采用多个分类器，同时将三个弱分类器作为一个强分类器，提高预测性能，但 SVM 难以适用于大规模数据集。因此近些年，卷积神经网络被应用于帧内编码单元快速划分算法中。文献[19]提出了一种使用卷积神经网络加速器的 VLSI 快速算法。文献[20]通过探索编码单元深度、量化参数和纹理复杂度的相关性，建立了一个分类的阈值模型。利用启发式方法，设计了三种不同的卷积神经网络结构，用于不同级别编码单元的训练，降低了编码复杂度，实现了更好的编码性能。文献[1]提出了一种基于 H.265/HEVC 编码深度图预测的快速块分区算法，使用深度图来表示一个编码树单元的块划分，该编码深度图通过卷积神经网络得到，摆脱了用于划分的递归率失真代价计算过程。

1.3 论文的主要内容和安排

本文利用深度学习技术优化视频编解码标准 H.265/HEVC 的帧内编码单元划分算法，近一步降低 H.265/HEVC 的编码复杂度。主要工作内容如下：

1. 分析传统的帧内编码单元划分原理，根据卷积神经网络的回归模型，提出一种全连接层优化的神经网络编码单元快速划分算法，通过加深网络层数，使得该卷积神经网络能够更好地发掘视频图像像素信息与帧内编码单元划分信息之间的关系，有效利用编

码单元整体信息，提高编码单元划分预测准确度。在网络全连接层加入归一化量化参数一起训练，使得卷积神经网络支持不同的量化参数。将提出的算法在 H.265/HEVC 编解码框架中进行性能评估验证，实验表明该算法可以降低 H.265/HEVC 编码复杂度，提高 H.265/HEVC 的编码效率。

2．利用帧内编码单元划分与卷积神经网络分类器的共性，根据卷积神经网络的分类模型，提出一种基于分类思想的帧内编码单元快速划分算法，利用神经网络设计三个分类器，以判断不同深度的编码单元是否向下一级划分，设计了连接三层神经网络的归并层，融合了整个编码树单元的信息特征，提高了神经网络准确性。引入小卷积核，减少该网络模型的参数、扩大感受野、提高网络性能，同时在网络全连接层加入归一化量化参数一起训练，使得卷积神经网络支持不同的量化参数。将提出的算法在 H.265/HEVC 编解码框架中进行性能评估验证，实验证明该算法提高了 H.265/HEVC 的编码效率。

全文的结构安排如下：

第一章主要对视频编解码 H.265/HEVC 的研究意义、研究背景和视频编解码的发展史进行阐述，之后对传统的帧内编码单元快速划分方法和基于深度学习的帧内编码单元快速划分方法进行总结，最后对论文的主要内容和安排做简单的阐述。

第二章介绍视频编解码标准 H.265/HEVC 的混合编码架构和深度学习基础知识。首先，介绍 H.265/HEVC 编码框架，阐述 H.265/HEVC 编码过程，然后对 H.265/HEVC 的各个模块涉及的技术进行介绍。由于本文主要研究帧内编码单元快速划分技术，所以进一步分析帧内编码单元划分原理。最后，对 ResNet 神经网络原理、损失函数、激励函数、优化算法等进行深入的研究，为后续的研究打下良好的基础。

第三章提出一种基于回归网络的编码单元划分算法。首先，根据视频编解码标准 H.265/HEVC 官方编解码器 HM 中帧内编码单元划分技术，分析编码单元划分影响因素。根据分析结果，构建包含视频亮度信息和编码树单元划分结构信息的预处理数据集，介绍基于回归网络的编码单元划分算法架构；最后，将提出的算法在 H.265/HEVC 编解码框架中进行性能评估，验证算法的有效性。

第四章提出一种基于分类网络的编码单元划分算法。首先，分析帧内编码单元划分方式与卷积神经网络分类器的共同特点，并在此基础提出基于深度学习的帧内编码单元快速划分分类网络；之后，给出卷积神经网络应用于 HM 编码单元划分函数中的具体算法；最后，将提出的算法在 H.265/HEVC 编解码框架中进行性能评估，验证算法的有效性。

第五章对本文的研究内容进一步总结，并分析了研究内容的局限性，最后对今后的研究方向进行展望。

第2章 H.265/HEVC 与深度学习技术基础

2.1 H.265/HEVC 编解码标准

H.265/HEVC 遵循混合编码框架：首先将视频分解为多帧图像，对每帧图像进行块划分的预处理操作，再把原始编码块信息与由帧间或帧内算法得到的预测值做差，得到对应的残差信号。将得到的残差数据经过变换、量化模块进行压缩。压缩后的数据一方面经过熵编码模块进行处理，得到最终输出比特流，另一方面经过反变换、反量化后得到残差值，将残差值与预测值相加得到重建值，随后通过环路滤波算法进行图像重建，将得到的图像作为参考帧。如图 2.1 所示，H.265/HEVC 视频编解码标准由熵编码、帧内预测、变换和量化以及帧间预测等模块组成，下面将对这些模块加以详尽说明。

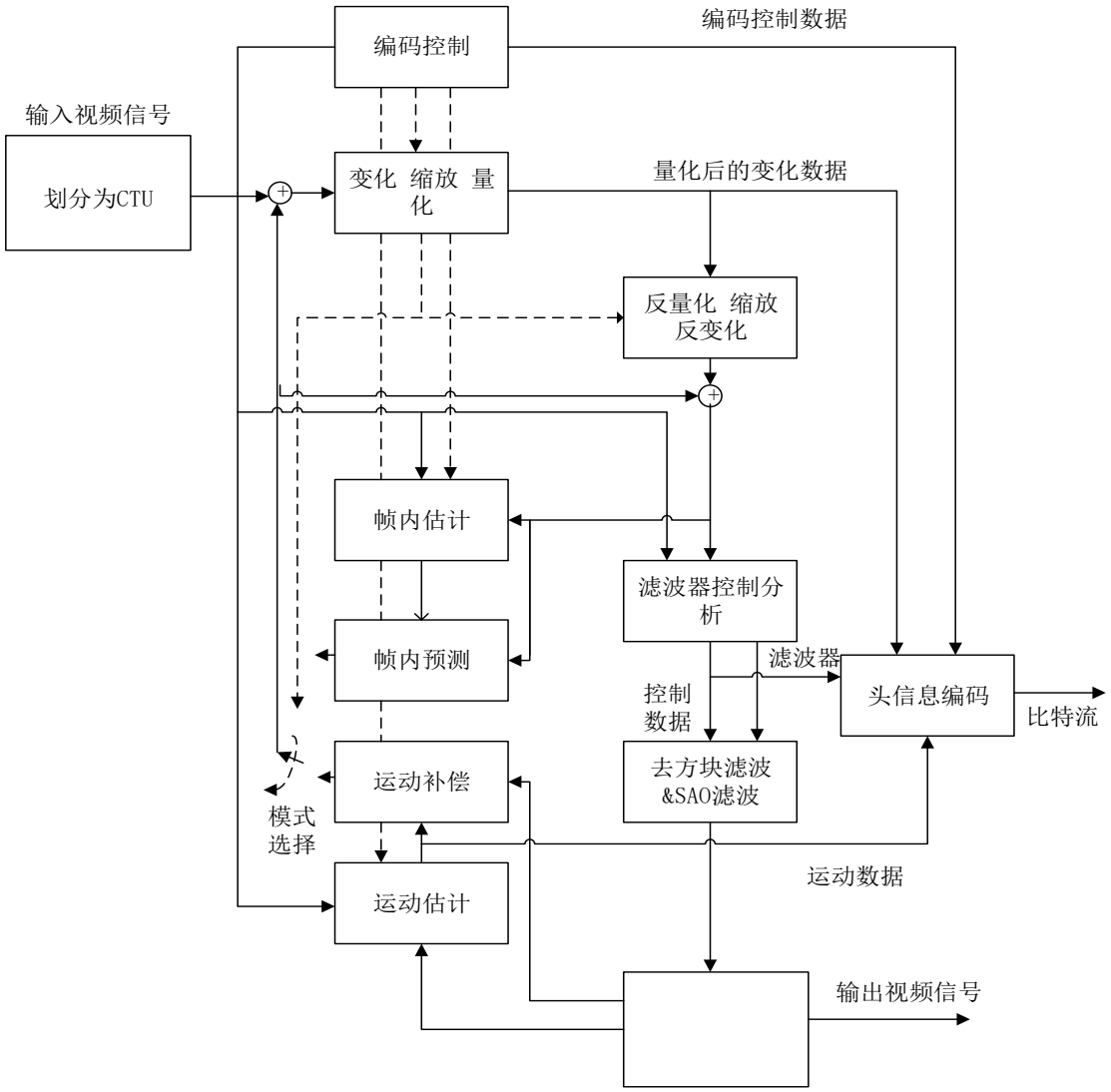


图 2.1 H.265/HEVC 编解码框架

2.1.1 预测编码

视频由许多帧按照时间顺序排列组成，每一帧都由 N×M 个像素组成，每个像素都有具体的数值。同一帧采集的图像像素之间存在着明显的空间关联性，而相邻帧中像素则存在较强的时间关联性，通过预测编码，能够有效地减少视频中的时间冗余和空间冗余。帧内预测编码利用同一帧中已编码的块来预测当前块，并将获得的残差信号作为后续输入，从而去除图像的空间冗余。

根据 H.265/HEVC 的规定，编码树单元 CTU（Coding Tree Unit）是编码器的基本处理单元，同时 H.265/HEVC 提出了变换单元 TU（Transform Unit）、预测单元 PU（Prediction Unit）和编码单元 CU 的概念。熵编码、帧内预测编码以及量化是在编码单元的基础上进行的，编码单元由色度信息、亮度信息和它们对应码流的语法语义组成，包括 64×64、32×32、16×16 和 8×8 四种尺寸。预测单元由编码单元进行四叉树划分得到，预测单元包含编码单元的所有预测模式和一切与预测有关的信息[21]。变换单元是变换和量化的基本单元，由编码单元进行四叉树划分得到，最大支持 32×32 大小的块，最小支持 4×4 大小的块。

此外，在帧内预测方向上，H.265/HEVC 为亮度定义了 35 种预测模式，这 35 种预测模式分别为 Planar 模式、33 种角度模式和 DC 模式，远远超过 H.264/AVC 的 17 种预测模式。具体流程可分为以下 3 个步骤：

1．检查当前变换单元参考像素的有效性，并使用对应的算法进行处理。

2．对参考像素进行滤波。

3．通过滤波处理后的参考像素，计算出当前变换单元的预测像素值，以便进行分析和预测。

在实际使用中，根据具体情况，可以直接略过滤波过程，进行步骤 3。对于步骤 1，参考像素的选取模式如图 2.2 所示，当前变换单元的尺寸为 N×N，共分为五个参考像素区域：左下（A）、左侧（B）、左上（C）、上方（D）和右上（E），一共 4N+1 个点，称之为 L 型像素。当变化单元参考区域部分像素信息丢失时，会用与信息缺失区域最接近区域的像素来填补，比如区域 E 尚未进行编码或处于片的边界时，其参考像素是不可用的，则区域 E 的像素用区域 D 最右侧的像素替代。当五个区域的像素均不存在时，用固定值进行填充。步骤 2 和步骤 3 针对不同的预测模式进行不同的处理，步骤 2 对不同尺寸的变换单元采用不同数量的模式进行滤波，步骤 3 根据预测模式采取对应的参考像素计算方式，可以实现对各种模式的有效操作，从而提高预测精度。

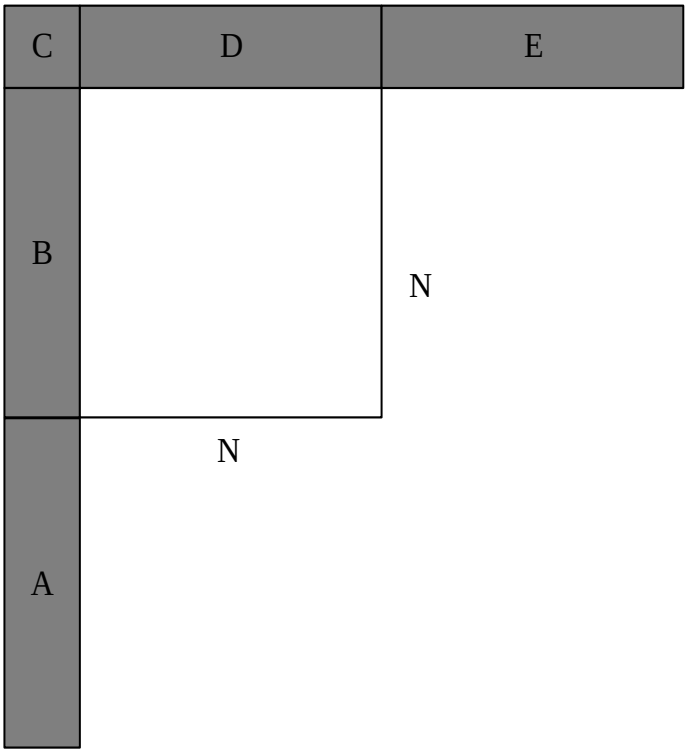


图 2.2 相邻参考像素位置

H.265/HEVC 采用拉格朗日率失真代价 RDO（Rate Distortion Optimization）作为评价预测模式优劣的标准。编码器遍历预测模式并计算拉格朗日率失真代价，具体计算如下式所示：

$$J = D + \lambda \cdot R \tag{2.1}$$

其中，D 代表帧内预测模式引起的失真，R 代表该帧内预测模式所需的比特数，λ 是拉格朗日乘子，由文献[22]的方法确定。率失真代价越小，编码的压缩性能越好。为了选择最合适的预测模式，编码器要遍历 35 种预测模式，导致编码器计算工作量极大。

帧间预测编码将重构后的图片当作参考帧，以预测后续帧的像素值。帧间预测编码获取每个块的运动信息，高效地消除了视频图像间的时间冗余，进而提升了视频编码和解码的效率。视频编解码标准包含三种帧类型：I 帧、B 帧和 P 帧。I 帧是独立关键帧，完全通过帧内预测编码进行视频压缩。B 帧是双向预测帧，它不仅可以参考前一帧的编码结果，还可以通过对后续帧的分析来进行预测，从而提高编码效率。P 帧是前向参考帧，运动信息等只与前面的帧有关。帧间预测编码包括运动补偿和运动估计两个步骤。在以块运动补偿为基础的视频编解码框架中，运动估计是其中一个关键步骤，也是一个耗费大量时间的过程。由于难以精确地将运动物体与背景分离开来，大多数运动估计算法都是基于像素值进行的。运动估计是在参考帧中找到与当前帧最匹配的像素块，然后利用运动矢量和预测方式，计算出当前帧的预测值[23]。帧间预测利用运动补偿来描述参考帧像素块与当前帧的像素块之间的差别，运动补偿对参考帧的每个像素块是怎样移动到当前帧中对应位置的进行了说明。

2.1.2 变换量化

变换编码把时间域上的信息转换为频率域上的信息，使原来分散的信息聚焦于较低的频率范围内，这能够有效地减少变换系数，进而提升压缩性能，最终实现去除冗余的目的。H.265/HEVC 支持 32×32、16×16、8×8、4×4 四种尺寸的变换单元，以满足不同视频内容的需要，最优划分方式可使用率失真优化准则来确定。H.265/HEVC 使用整数离散余弦变换 DCT（Discrete Cosine Transform），基本变换公式为：

$$X(k,l)=C(k)C(l)\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}x(m,n)\cos[\frac{(2m+1)k\pi}{2N}]\cos[\frac{(2m+1)l\pi}{2N}] \tag{2.2}$$
$$k,l=0,1\cdots,N-1$$

其中

$$C(k)=C(l)=\begin{cases}\sqrt{\frac{1}{N}}, & k,l=0\\ \sqrt{\frac{2}{N}}, & \text{其他}\end{cases} \tag{2.3}$$

其逆变换如下：

$$x(m,n)=\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}C(k)C(l)X(k,l)\cos[\frac{(2m+1)k\pi}{2N}]\cos[\frac{(2n+1)l\pi}{2N}] \tag{2.4}$$
$$m,n=0,1\cdots,N-1$$

当灰度值变动较慢时，DCT 算法能够将部分能量聚集在低频系数上，当图像中含有较多细节纹理信息时，则会使能量散布在高频区域。大多数视频图像中各元素幅值差小，相邻像素相关性强，含有更多的低频信号，因此，利用人眼对图像高频细节的不敏感性，对高能量的低频系数加以精确的计算，对低能量的高频系数进行粗略的计算，通过这种方式，能够更好地压缩图像，避免显著降低主观质量[24]。

图 2.3 H.265/HEVC 变换量化过程

图 2.3 介绍了 H.265/HEVC 的变换量化过程。在编码器中，残差数据经过变换编码后，变化范围大，量化编码将数据划分为多个区间，每个区间选择一个值代表该区间的所有值，它将连续值映射为离散数据，能够有效减少信号取值范围。因此，视频经过量化编码后能够获得更好的压缩性能。H.265/HEVC 使用的传统标量量化方法公式如下：

$$l_i=floor(\frac{c_i}{Q_s}+f) \tag{2.5}$$

其中，c_i 是整数余弦变换系数，Q_s 是量化步长，l_i 表示量化值，floor()是向下取整函数。

在视频编码中，量化参数是非常重要的参数，它直接影响着视频的编码比特率。在某些传输速率受限的视频应用场合，灵活地控制量化参数使得编码速率尽量接近给定速率尤为重要。为此，H.265/HEVC 视频编解码标准规定了 52 个量化参数 QP（Quantization Parameter），与 52 个量化步长一一对应，对应关系如下：

$$Q_s = 2^{(QP-4)/6} \tag{2.6}$$

量化步长越大，量化强度和图像损失越高，码率越小；量化步长越小，量化强度和图像损失越低，码率越大[25]。

2.1.3 熵编码

熵编码是无损编码方式，熵编码模块将编码控制、帧内预测、变换量化系数和帧间预测等数据编码为适合进行存储或传输的码流。H.265/HEVC 使用基于上下文的自适应二进制算术编码 CABAC（Context-based Adaptive Binary Arithmetic Coding）作为主要熵编码方法。

如图 2.4 所示，CABAC 编码包括三个主要步骤：二进制化、上下文建模和二进制算数编码。二进制化是将某个非二进制结构转化为一种二进制序列，即二元流，它可以用来表示一种复杂的数据结构。假设使用的语法是二进制的，那么二进制化的处理过程就会被省略，信息将经过一段旁路传输到下一层，即二进制算术编码。二进制算术编码有两个方式：普通方式和旁路方式。在普通编码方式中，语法元素的二元位会按照一定的顺序被上下文模型器捕捉和管理，以便更好地表达信息，编码器通过已编码的二元位的值为当前的二元位分派对应的概率模型，这一过程被称为上下文建模。将这些二元值和编码器分派的概率模型一同输入二进制算数编码器中进行编码。编码器可以按照二元值自动调整上下文模型，从而实现自适应编码。此外，旁路编码模式可以有效地提高编码的速度，因为它不需要为各个二元位分配特殊的概率模型，而是直接使用一种简易的旁路编码器来处理输入的二元值，从而大大提高了编码的效率。

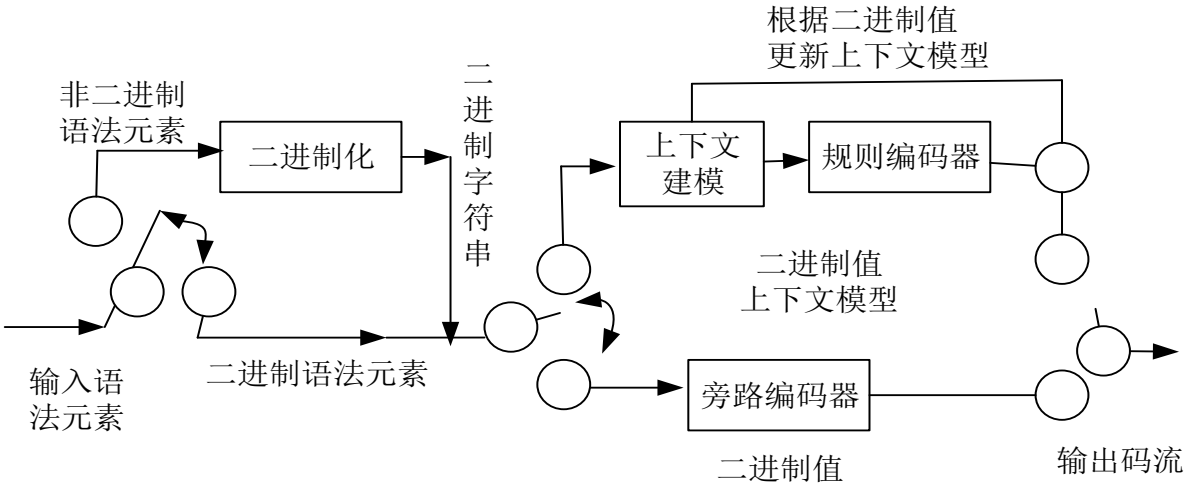


图 2.4 CABAC 流程

2.1.4 环路滤波

H.265/HEVC 是基于宏块的视频编解码技术，视频经过预测编码、变换和量化、反变换和反量化等过程，不可避免的会遇到振铃效应、方块效应、马赛克等失真问题。图像经过环路滤波处理进一步减少失真，重构后的图像才能作为后续帧的参考图像。H.265/HEVC 采用自适应的去方块滤波（Deblocking Filter）技术和像素样点自适应补偿技术 SAO（Sample Adaptive Offset）改善图像质量，从而提高视频视觉质量。样点自适应补偿技术解决了高频信息丢失的问题，SAO 技术将重建像素分为不同的类别，根据像素类别采用相对应的补偿值，以减少图像失真[26]，图 2.5 展示了 SAO 的整体流程。去方块滤波技术解决了图像编码块边界不连续的问题。图 2.6 显示了去方块滤波流程，去方块滤波算法包括滤波操作和滤波决策两个步骤。滤波决策根据视频内容决定是否进行滤波以及选择滤波强度，滤波操作针对不同的滤波强度使用不同的方式修改图像像素。

图 2.5 SAO 流程

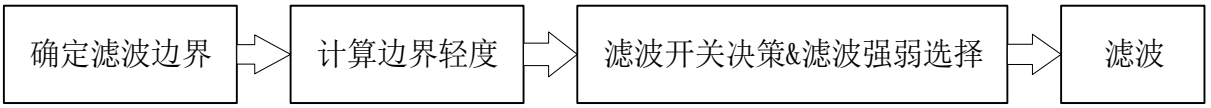


图 2.6 去方块滤波

2.2 编码单元划分原理

在视频编码过程中，采用较小的块可以很好地体现纹理细节，较大的块可有效提高编码效率。宏块的大小如何确定是视频编解码标准需要解决的问题。为此 H.265/HEVC 提出了编码树单元和编码单元[27]的概念，并提出了四叉树划分算法。在 H.265/HEVC 中，一帧图像在编码前要被划分为若干个不重叠的尺寸为 64×64 的编码树单元，远远大于 H.264/AVC 规定的大小为 16×16 的宏块。图 2.7 是 H.265/HEVC 帧内编码单元划分结构示意图，在标准编码器中，编码单元采用四叉树算法递归划分，从最基本 64×64 编码树单元开始，一个编码树单元可能不会被划分，也可能被分割成四个子编码单元。每个子编码单元又可以选择是否被分成四个更小的子编码单元，以此类推，直到编码单元被分成 8×8 的最小尺寸为止。当深度为 n 的编码单元被分割为 4 个子编码单元时，子编码单元深度会加 1，因此编码单元 64×64 到 8×8，对应着深度 0 到 3。为了选择最合适的编码单元划分方式，编码器使用率失真代价作为指标来决定编码单元的划分，其计算公式为式（2.1）。编码单元最大尺寸是 64×64，最小尺寸是 8×8，多种编码单元尺寸为 H.265/HEVC 提供灵活的块分割方式。

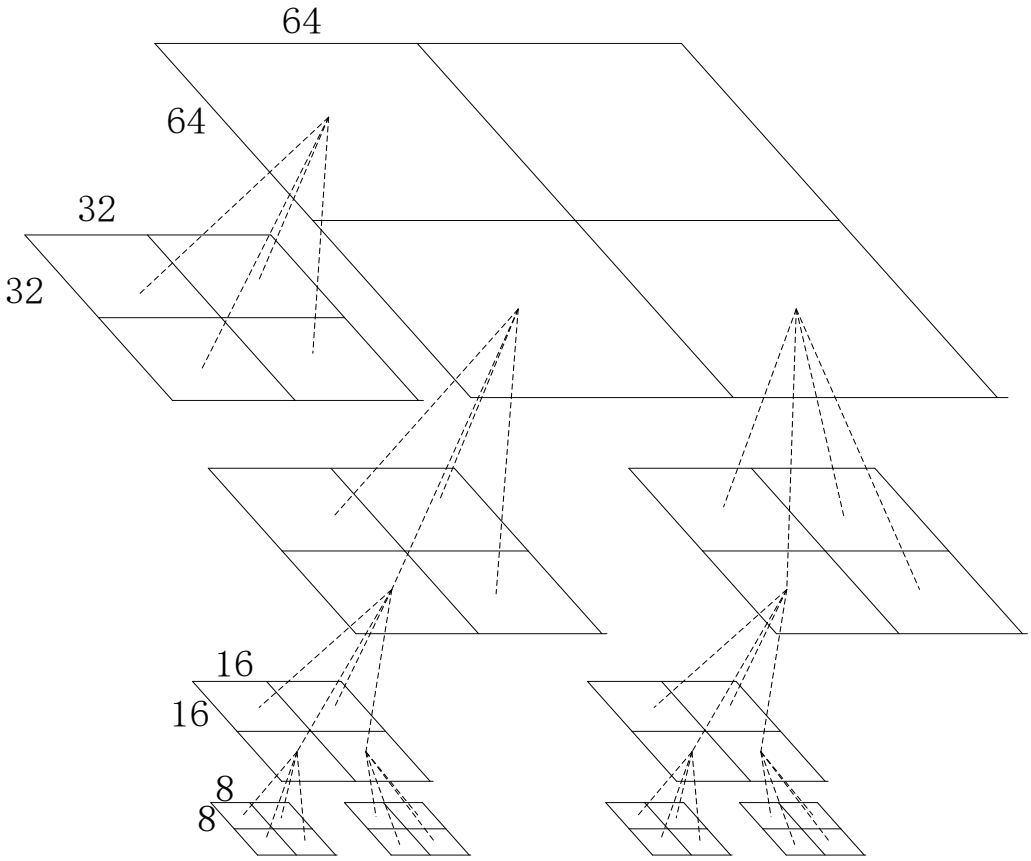


图 2.7 帧内编码单元划分

相对于 H.264/AVC 的宏块划分方法，这种编码树单元和编码单元的表示方法有以下优点：

1．编码单元最大尺寸远大于常规的 16×16 宏块，对背景简单的图像，用大尺寸的编码单元可以减少编码占用的比特数，提升视频编码性能，对具有复杂纹理的图像，用小尺寸的编码单元可以提高图像质量。

2．根据视频图像内容、视频分辨率和视频应用领域，可自适应地选择合适地编码单元尺寸和深度。

3．这些单元都被称为编码单元，块的划分更加统一和清晰，编码单元结构可通过对应的划分标记、最大编码深度和编码单元尺寸来描述。

H.265/HEVC 的官方编码器 HM 中，帧内编码单元递归划分的具体流程如图 2.8 所示：

1．在深度为 0 时，编码器对 64×64 编码单元进行编码，并将最优预测模式、运动信息及率失真代价等编码信息进行记录；

2．将 64×64 编码单元划分为四个深度为 1 的 32×32 子编码单元。并将四个子编码的最优预测模式、运动信息及率失真代价等编码信息分别进行记录；

3．将 32×32 编码单元划分为四个深度为 2 的 16×16 子编码单元。并将四个子编码的最优预测模式、运动信息及率失真代价等编码信息分别进行记录，继续向下划分；

4．对四个深度为 3 的 8×8 子编码单元同样进行循环编码，并将最优预测模式、运动信息及率失真代价等编码信息进行记录。然后进行率失真代价比较，选择最优的编码单元划分模式。

5．通过比较 16×16 尺寸编码单元的率失真代价和四个 8×8 尺寸的子编码单元率失真代价之和，确定是否将深度为 2 的编码单元划分为 4 个深度为 3 的子编码单元。

6．在完成所有 8×8 的子编码单元与相应的 16×16 编码单元比较后，通过比较 32×32 尺寸编码单元的率失真代价和四个 16×16 尺寸的子编码单元率失真代价之和，确定是否将深度为 1 的编码单元划分为 4 个深度为 2 的子编码单元。

7．在完成所有 16×16 的子编码单元与相应的 32×32 编码单元比较后，通过比较 64×64 尺寸编码单元的率失真代价和四个 32×32 尺寸的子编码单元率失真代价之和，确定是否将深度为 0 的编码单元划分为 4 个深度为 1 的子编码单元。

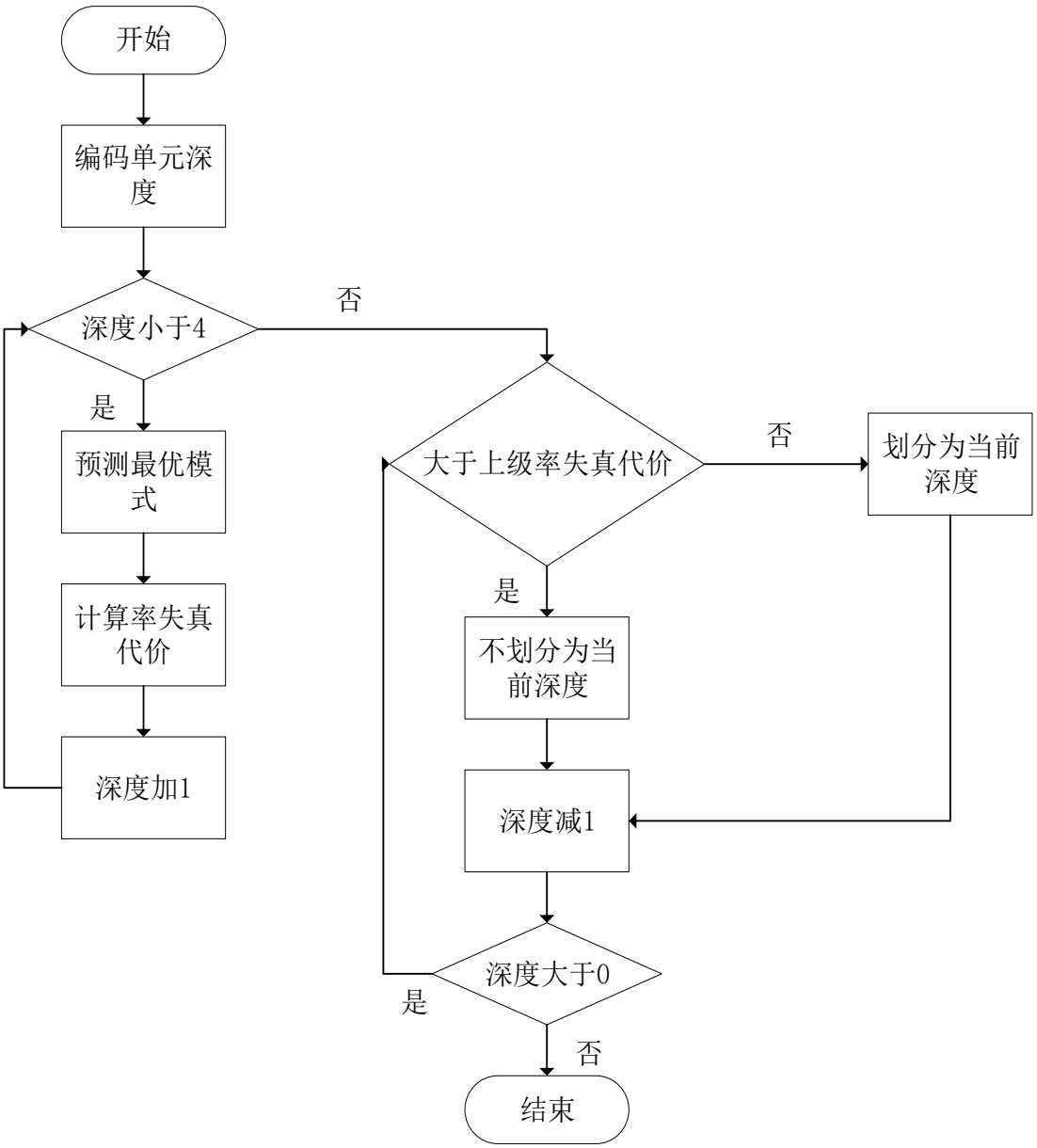


图 2.8 HM 中编码单元划分流程

综上可知，对于一个 64×64 大小的编码树单元，编码器遍历 4 种尺寸的编码单元，共需要进行 85 次编码单元运算，1935 次残差变换绝对值和 SATD（Sum of Absolute Transformed Difference）代价运算和至少 2623 次率失真代价运算，计算量巨大[28]。因此，有必要对帧内编码单元划分技术进行研究，以降低视频编解码标准的算法复杂度。

2.3 深度学习理论基础

深度学习（Deep Learning）利用计算机从经验中进行学习，使用层层迭进的架构来解决问题，每一层概念通过与更简单的概念之间的关系来定义。让计算机从经验获取知识，可以避免让人类固定化地给每个概念下定义。一层一层概念的堆叠，就会得到一张很“深”的图。

深度学习具体框图如图 2.9 所示，它可以从大量的数据中提炼出复杂的模式，从而学习样本的内在规律。目前，许多不同的神经网络架构，包括深度神经网络、递归神经网络、深度置信网络，都可以更准确地学习各种复杂的问题。随着技术的不断发展，深度学习已经成为一种重要的驱动力，它不仅可以提供高效的决策支持，还可以实现高精度的数据处理、高效的自动化和模块化。

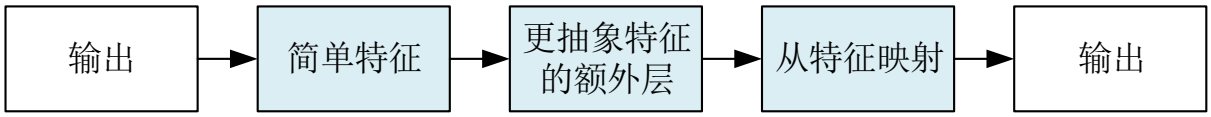


图 2.9 深度学习过程

2.3.1 深度神经网络

神经网络 NN（Neural Networks）也被称为人工神经网络 ANN（Artificial Neural Networks），是一种自适应系统，它通过模拟大脑神经突触的结构来处理信息，实现了自动化和智能化。神经网络可以从海量数据中提取出有用的特征，并利用这些特征来训练模型、分类数据和预测未来事件。神经网络适合于建模非线性关系，通常用来进行模式识别，并对语音、视觉和控制系统中的对象或信号进行分类。神经网络是一种重要的机器学习技术，它可以解决预测复杂和需要深入理解的问题。

神经网络灵感来源于人脑中的神经元，通过模拟人脑分层结构中的互连节点或神经元进行学习。神经网络由三个部分组成：输入、输出和运算。输入单元相当于神经元的树突，输出单元相当于神经元的轴突，而运算单元则相当于神经元的核心[29]。如图 2.10 所示，在简单的神经元模型中，每一条有向箭头线被称之为连结，而每一条连结上都有某个值，也就是权值或权重。优秀的神经网络训练算法旨在通过调整权重值来提升模型的性能，从而达到最佳的学习效果。

使用 a 来表示输入，用 w 来表示权值，$a*w$ 表示加权后的信号，g 表示对加权后的信号进行函数计算，Z 是输出，可以表示为：

$$Z = g(a_1 \cdot W_1 + a_2 \cdot W_2 + a_3 \cdot W_3) \tag{2.7}$$

使用矩阵表示为 *Z=g(W*A)*。

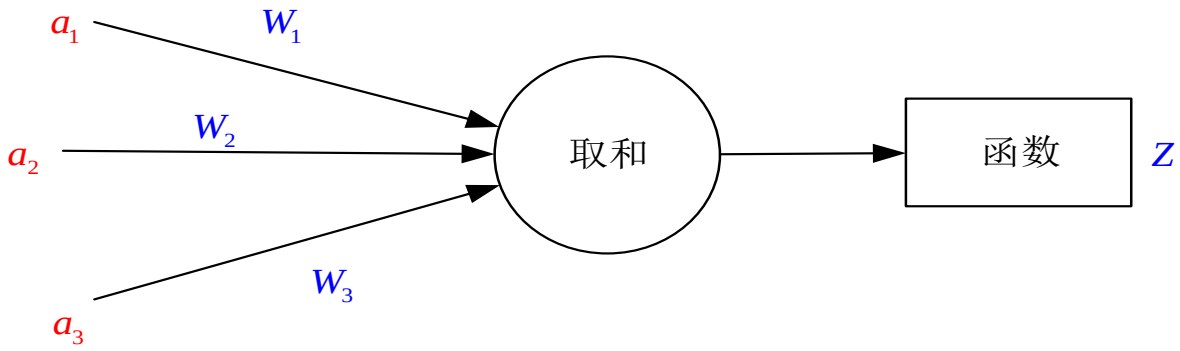


图 2.10 神经元模型

神经元需要存储输出并将输出传递给下一层，作为下一层的输入。一个简单的两层神经网络如图 2.11 所示，包括输入层、输出层和中间层。中间层输出使用矩阵表示为$A^{(2)}=g_1[W^{(1)}*A^{(1)}]$，输出层输出使用矩阵表示为$Z=g[W^{(2)}*A^{(2)}]$。多层神经网络是一种复杂的神经网络结构，通过在两层神经网络的输出层之后增加层次，可以将其转换为更复杂的模型，从而搭建更高效的神经网络结构。多层神经网络可以学习比较复杂的问题，模型有很强的表达能力，训练和测试的计算并行性好，可应用于分布式系统。

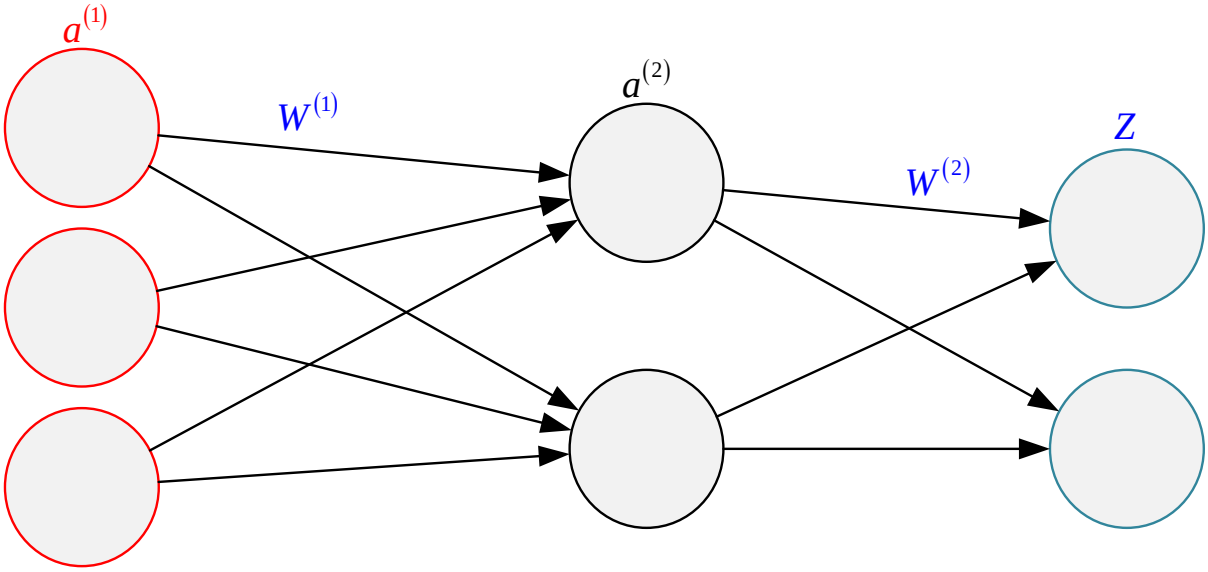


图 2.11 简单的两层神经网络

神经网络技术发展至今，主要有三种类别：反馈神经网络、前馈神经网络和图神经网络。前馈神经网络（Feedforward Neural Network）具有更加完善的结构，由输入层-隐藏层-输出层构成，数据从输入层开始，通过一系列神经元的处理，最终传达到输出层，每一层都会接收到前一层的数据，它们之间存在着相互作用，从而达到对数据有效处理的目的。常用的模型结构有：卷积神经网络、全连接神经网络。与前馈神经网络相比，反馈神经网络（Feedback Neural Network）存在一个从输出层到输入层的反馈回路，它具有很强的联想记忆能力和优化计算能力。常用的模型结构有：循环神经网络、Hopfield

网络、玻尔兹曼机等。图神经网络将图像处理技术与人工智能相结合，它能够在图像上进行端对端的计算，并且能够保留图像的结构信息。这种方法能够有效地提高图像处理的准确性和效率。图神经网络可以通过图卷积、图自编码、图生成、图循环和图注意力等多种模型来实现复杂的任务。下面就本文涉及的卷积神经网络进行介绍。

（1）卷积神经网络

卷积神经网络属于前馈神经网络，它由输入层、卷积层、池化层和全连结层构成，具有深度学习能力和模型优化能力，被广泛应用于数据处理、模式识别、生物信息监测与分析等领域，并在这些领域取得了显著的成果。感受野和权值共享是卷积神经网络的重要特性。

感受野将输入图像中的像素点投射到特定的地方，使特征图能够更好地反映输入图像的特点，提高模型的准确性和可靠性。当卷积尺寸较小时，网络参数会减少，网络深度会增大，感知野也会扩大，网络性能提高；如果感受野与目标尺寸不匹配，模型收敛会变得艰难，影响学习效果。感受野尺寸的计算是从最后一层特征图开始，采取从下往上的计算方法，即先计算最深层在前一层上的感受野，以此类推逐层传递到第一层。感受野的计算公式为：

$$E_i = (E_{i+1} - 1) \times P + Z \tag{2.8}$$

其中，E_i 代表第 i 层感受野，E_{i+1} 代表第 $i+1$ 层的感受野，P 表示第 i 层卷积步长，Z 是第 i 层卷积核的大小。感受野计算有以下几点规律：感受野的计算不考虑填充的大小；卷积神经网络最后一层感受野的值与卷积核尺寸相等。

权值共享的概念在卷积层提出，同一个卷积核对整张图片进行卷积运算，该卷积核内的权重参数被整张图共享，并不会因为位置因素影响权重值。每个卷积核只关注一个数据特性，所以深度神经网络需要多个卷积核提取多个特征，以便后续进行识别。权值共享减少了卷积神经网络中参数的个数，大大降低网络训练难度，减少网络训练时间。

（2）损失函数

深度学习本质上是给定 x 值与 y 值，学习 x 到 y 的对应关系，给定一个不在训练集内的 x，根据学到的规律，得到对应的 y 值。为了衡量预测值与真实值之间的差距，损失函数（Loss function）应运而生。在深度神经网络中，损失函数是一个用来计算深度神经网络模型预测数据与实际值差异的重要参数。损失函数越小，说明网络训练效果越好。下面介绍在深度学习中常用的损失函数。

均方误差 MSE（Mean Square Error）是一种常见的损失函数，它可以用来估计深度学习回归任务中的误差，也被称为 L2 Loss。

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{2.9}$$

其中，y_i 代表第 i 个样本的真实值，$\hat{y}_i$ 代表第 i 个样本的预测值，N 代表采样总个数。MSE 只能用来评估数据发生变化的情况，MSE 的值越小，说明预测模型的准确性越高。

MSE 具有快速收敛的特点，能够给层次赋予合适的权重，使得梯度迭代更新的方式更为准确。然而，它对异常值非常敏感，梯度迭代更新的方向极易受到外部因素的影响，缺乏鲁棒性[30]。

平均绝对误差损失 MAE（Mean Absolute Error Loss）也常被用于深度学习回归任务中，被称为 L1 Loss。其计算公式如下：

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{2.10}$$

其中，y_i是第 i 个数据的真实值，$\hat{y}_i$ 代表第 i 个数据的预测值，N 代表数据总个数。MAE 对离群点或者异常值更具有鲁棒性，但是它在零点处的导数不连续，导致求解效率低下，收敛速度也较慢，不利于网络的学习[31]。

根据上节介绍可知，L2 loss 收敛快但鲁棒性差，L1 loss 鲁棒性强但收敛慢。因此，Huber Loss 将上述损失函数的优点结合起来，在误差接近零的时候使用 L2 loss，误差较大时使用 L1 loss，公式为：

$$L_\delta = \begin{cases} 0.5(y-\hat{y})^2 & |y-\hat{y}| \le \delta \\ \delta|y-\hat{y}| - 0.5\delta^2 & otherwise \end{cases} \tag{2.11}$$

其中，y 代表真实值，$\hat{y}$ 代表预测值，δ 是 Huber Loss 的一个超参数，它是连接 MSE 和 MAE 损失函数的因子。Huber Loss 在误差接近 0 时使用 MSE，损失函数可导，梯度更为平稳；在误差较大时使用 MAE 可以减少异常值的负面影响，使训练更加稳定。它的缺点是需要额外地设置一个超参数。

合页损失 Hinge Loss 是一种二分类损失函数，适用于向量机 SVM（Support Vector Machine）模型，通常被用于最大间隔算法(maximum-margin)。Hinge Loss 的公式如下：

$$Loss = \max(0, 1 - yf) \tag{2.12}$$

其中，y 代表真实值，f 代表预测值。合页损失函数的目的是找到一个决策边界，以确保所有数据点都能被准确、可靠地分类，并且只有当确信度达到一定程度时，损失函数才会降至零。

（3）反向传播

神经网络的训练过程为：首先，利用随机赋值来获取预测结果；其次，利用损失函数来比较预期值与现实值之间的差别；最后，利用反向传播计算不断更新神经元参数，以获得更准确的预测结果。循环以上步骤，直到预测结果和真实结果几乎相同时，结束训练。前向传播是一种从输入到输出的神经网络模型，它能够统计和存储每个层次的结果。而反向传播则是一种改进神经网络参数的计算方式，它从输出层开始，逐层遍历网络系统，并统计每个中间变量和参数的梯度，以便更好地优化网络。梯度下降法寻找损失函数的最小值以更新权重参数，控制迭代更新方向，最终使模型收敛。下面介绍一下反向传播计算流程和原理。

图 2.8 是一个简单的神经网络计算流程图。从左到右是前向传播，输入批量数据 A，计算公式为：

$$\begin{aligned} Z &= WA + b \\ A &= g(Z) \end{aligned} \tag{2.13}$$

其中，W 是权重矩阵，b 是偏移项矩阵，$g()$是激励函数，Z 是输出矩阵。反向传播目的是寻找损失函数的最小值，则在反向传播中要计算权值参数的梯度和偏移向的梯度，以确定一个下降方向。假设损失函数为$L(\theta)$，学习率为η，对单个数据的权重进行更新：

$$\begin{aligned} w_i &= w_i - \eta \cdot \frac{\partial L}{\partial w_i} \\ \frac{\partial L}{\partial w_i} &= \frac{\partial L}{\partial z} \cdot \frac{\partial z}{\partial w_i} \end{aligned} \tag{2.14}$$

对偏移项进行更新

$$\begin{aligned} b &= b - \eta \cdot \frac{\partial L}{\partial b} \\ \frac{\partial L}{\partial b} &= \frac{\partial L}{\partial z} \cdot \frac{\partial z}{\partial b} \end{aligned} \tag{2.15}$$

根据链导法则和式（2.13），式（2.14），从后往前依次计算每一层新的参数值，这个过程就是反向传播。

（4）优化算法

卷积神经网络的目的是搜索一组权重参数，使得定义的损失函数值最小。其实质是改进训练方法，使损失函数降到最低。下面介绍几个深度学习常用的优化算法。

随机梯度下降法 SGD（Stochastic gradient descent）每次使用一个或少量样本进行参数更新。经典的梯度下降每次加载整个数据集计算梯度，需要花费大量的时间和内存，无法将其应用到大数据集中；而随机梯度下降法为了降低内存和时间开销，放弃了对准确度的追求，每次计算梯度只用一个样本，单个样本更新参数大大加快了收敛速度，适用于源源不断的在线更新。随着参数的不断更新，损失函数会出现不同程度的波动，从而可能发现新的局部最小值，但是这种频繁的变化也可能导致超调量的出现。

动量（Momentum）梯度下降法中梯度的更新方向与上一次更新方向和当前负梯度方向有关。动量梯度法加快了下降，它在下降的方向上不断累积动量，使得下降的速度变快。其计算公式如下:

$$\begin{aligned} v_t &= \gamma v_{t-1} + \eta \nabla L(\theta) \\ \theta_{t+1} &= \theta_t - v_t \end{aligned} \tag{2.16}$$

其中，θ是给定待优化的模型参数，η是学习率，γ 经验值是 0.9。公式包含两部分，一个是学习率乘以当前的梯度，另一个是带衰减的前一次梯度更新。与随机梯度下降相比，动量方法收敛更快。

自适应时刻估计方法 Adam（Adaptive Moment Estimation）算法，能够根据参数的

特征，计算出最佳的自适应学习率，从而提高模型的准确性。Adam 算法包括惯性保持和环境感知两方面。计算梯度的一阶矩即过去梯度与当前梯度的平均可以反映出惯性保持思想；而计算梯度的二阶数即过去梯度的平方与当前梯度的平方的平均则可以体现出环境感知能力。Adam 的改进措施使神经网络的训练更具稳定性。其计算公式为：

$$\begin{aligned} \mathrm{m}_t &= \eta[\beta_1 m_{t-1} + (1-\beta) g_t] \\ v_t &= \beta_2 v_{t-1} + (1-\beta_2) \cdot diag(g_t^{\,2}) \end{aligned} \tag{2.17}$$

对一阶动量和二阶动量进行修改：

$$\begin{aligned} \hat{m}_t &= \frac{m_t}{1-\beta_1^t} \\ v_t &= \frac{v_t}{1-\beta_2^t} \end{aligned} \tag{2.18}$$

最终获得更新参数：

$$\theta_{t+1} = \theta_t - \frac{1}{\sqrt{\hat{v}_t} + \varepsilon} \hat{m}_t \tag{2.19}$$

（5）激活函数

激活函数是神经网络架构的关键一环，在不使用激活函数的情况下，每一层得到的都只是矩阵的乘积，它能将输入输出拟合为非线性关系，使神经网络能够解决具有实际意义的问题。激活函数可以将当前特征空间转换为更加精确的映射，从而使得数据更容易被准确地分类[32]。下面介绍几种常用的激活函数。

Sigmoid 函数是一种广泛应用的激活函数，它可以用来解决二分类的问题。Sigmoid 函数的数学表达式如下：

$$f(x) = \frac{1}{1+e^{-x}} \tag{2.20}$$

函数值域为（0,1）。Sigmoid 激活函数最大的优点在于求导方便，可对数据进行压缩，但是如图 2.12 所示，Sigmoid 函数不是以零为中心，而是随着网络层数的增加而发生变化，每一个神经元均会接受上一个神经元的非零均值输入和输出，这种变化会破坏原有数据信息的分布，从而严重影响整体网络系统对数据分布的拟合。

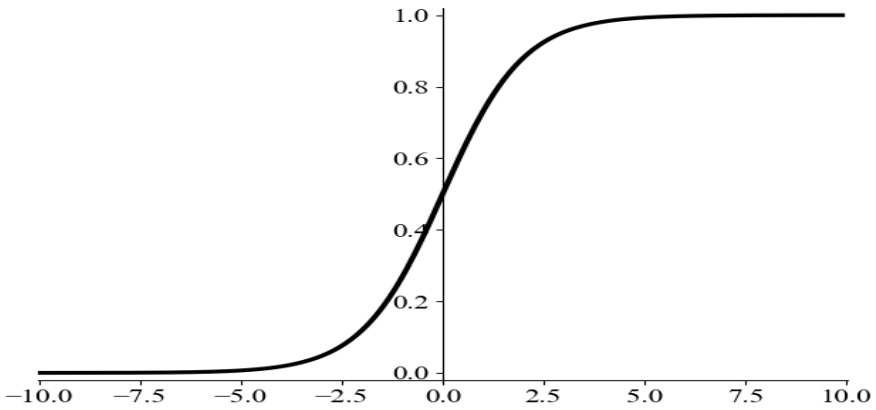


图 2.12 Sigmoid 函数

Tanh 函数将变量映射到[-1,1]区间，公式表达式如下：

$$f(x)=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{2.21}$$

如图 2.13 所示，Tanh 函数了有效地解决零点不对称的问题，它具备 Sigmoid 的优势，如平滑性、容易求导等，但也存有一定的缺陷，Tanh 激活函数运算量较大，梯度消失的现象获得了一定程度的缓解，但是无法彻底解决这个难题。

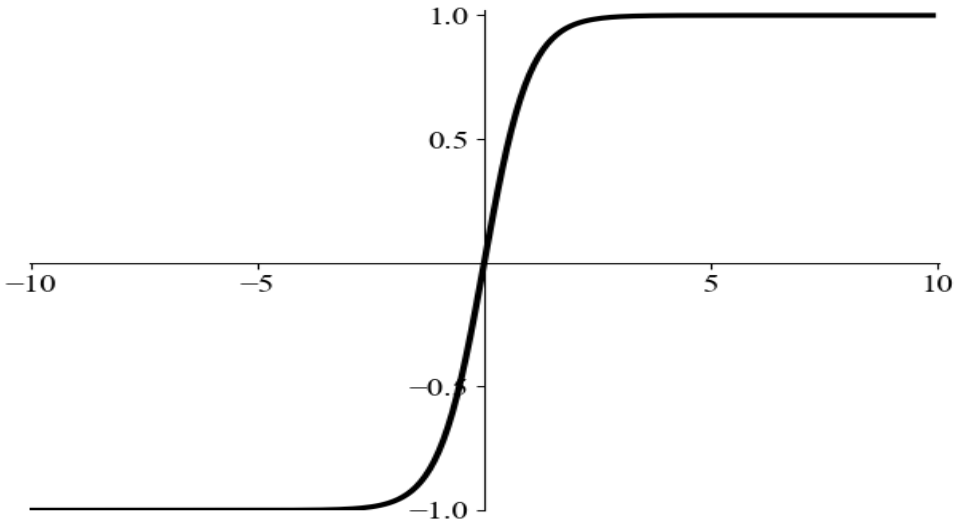


图 2.13 Tanh 函数

ReLU 激活函数可以提供一种更加高效的学习方式，因为它可以在输入超出阈值时，通过正向梯度的学习，激活神经元，此过程不需要复杂的数学运算，从而大大提高了计算速度。如图 2.14 所示，当输入为负值时，ReLU 的学习效率会大幅下降，以至于神经元完全失效，梯度为零，从而导致权重无法得到有效更新，使其在接下来的训练过程中失去作用。ReLU 函数的定义为：

$$\mathrm{Re}\,LU(x)=\begin{cases}x, & if \quad x>0\\ 0, & if \quad x\le 0\end{cases} \tag{2.22}$$

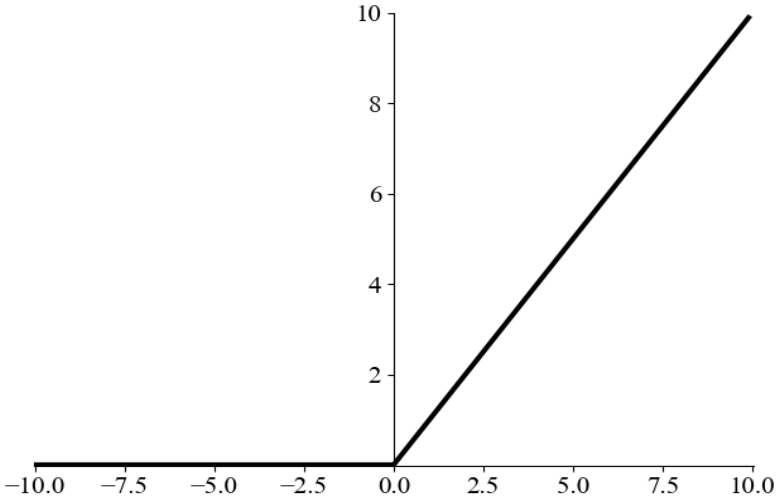


图 2.14 ReLU 函数

ReLU 函数具有以下优点：ReLu 函数具有线性和非饱和的特点，使得它能够快速收敛；Sigmoid 函数和 Tanh 函数包含许多非常复杂的运算，ReLU 则能较容易地实现；有效缓解了梯度消失的问题[33]，在没有监督预训练的情况下，ReLU 也能表现出良好的性能。ReLU 函数缺点在于：ReLU 的输出不是以零为中心的；一些神经元可能永远保持静止状态，从而使得相关的参数无法得到更新。

2.3.2 ResNet 网络

ResNet 是何凯明团队于 2015 年提出，并在当年获得 ImageNet 竞赛分类任务第一名，目标检测第一名。相比以往卷积神经网络，ResNet 网络存在以下特点：网络较瘦，控制了参数数量；具有清晰的层次结构，并将特征图数按层次逐步递增，以确保输出特征表示能力；使用批量归一化 BN（BatchNorm）和全局平均池化进行正则化，加快了训练速度；层数较高时减少了 3×3 卷积个数，并用 1×1 卷积控制了 3×3 卷积的输入输出特征图数量，称这种结构为“瓶颈”(Bottleneck)。

从理论上讲，网络层数加深，神经网络能够提取更多特征，网络学习能力越强。但实验发现一味地堆叠神经网络层数会出现退化问题（Degradation Problem）：网络深度增加时，网络准确度反而不如浅层神经网络[34]。随着网络层数的增加，会出现随机梯度消失或梯度爆炸的问题，这些问题使模型难以收敛，导致神经网络学习效果变差。文献[35]基于 CIFAR-10 数据集分别训练了 20 层和 56 层的网络，结果如图 2.15 所示（图源自文献[35]），20 层网络的训练误差和测试误差小于 56 层网络，作者提出网络在恒等映射上出现问题。针对以上问题，ResNet 神经网络提出残差结构的概念。

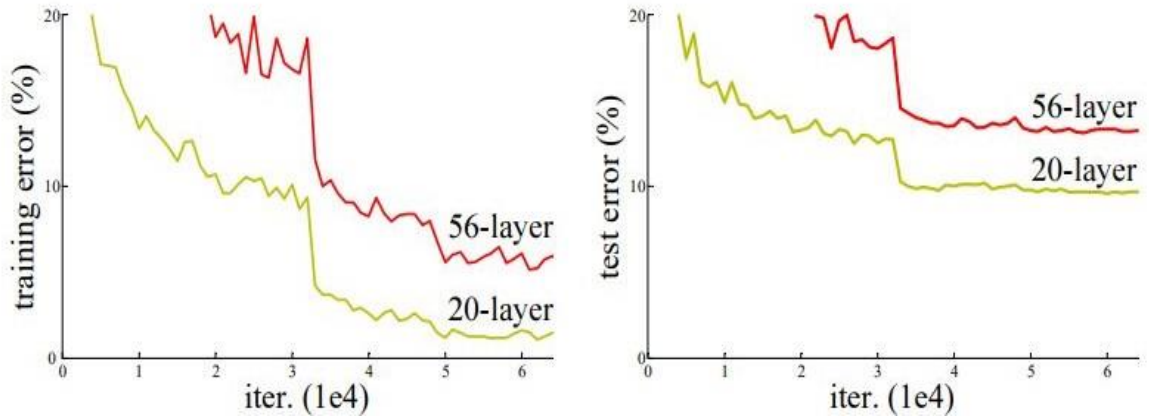


图 2.15 20 层和 56 层网络训练结果

恒等映射就是输入 x 经过某一个函数，输出还是 x 本身。如图 2.16 右侧所示，需要调整其内部参数，使得输入 x 经过学习后的最终输出 F(x)等于 x，即实现恒等映射 F(x)=x。但是卷积网络很难调整其参数完美地实现 F(x)=x。图 2.16 左侧即为残差结构，卷积层通过添加输入直接连接到输出的结构，使卷积网络的输出变成 F(x)+x。此时网络调整其内部参数使 F(x)+x=x，就是直接令其内部的所有参数为 0，使得 F(x)=0，F(x)+x=x 就变成

了 0+x=x，输出等于输入，完美地完成了恒等映射。因此 ResNet 网络减缓了深度退化带来的影响。

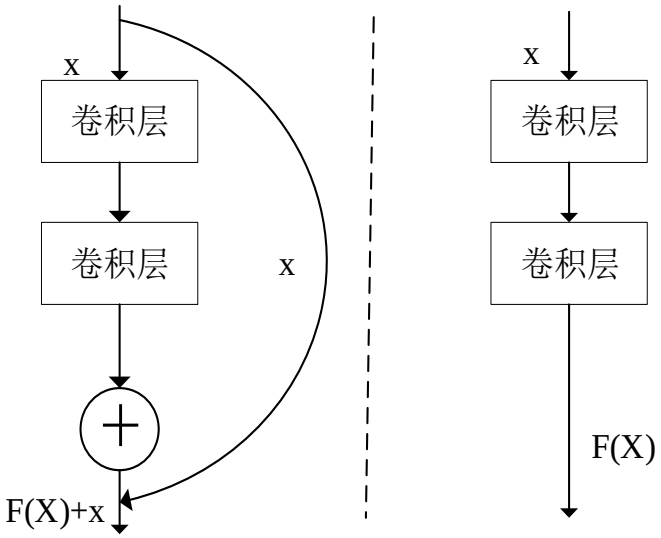


图 2.16 网络结构对比

第3章 CuprNet 网络编码单元划分算法

帧内编码单元划分是 H.265/HEVC 视频编解码框架的重要组成部分。为了提高视频编码的效率，降低视频编解码标准编码复杂度，提出了一种基于 ResNet18 网络的全连接层优化的帧内编码单元快速划分回归网络，并将网络应用于 H.265/HEVC 官方编码软件 HM 中，验证了算法的有效性。

3.1 编码单元划分影响因素分析

一般来说，帧内编码树单元的划分结构与视频图像的纹理复杂度有关[36]。对于平坦区域，大尺寸的编码单元就能够反映该区域的信息。因此，在对背景单一，纹理简单的视频进行编码单元划分时，编码器倾向于选择深度小的编码单元对图像帧进行编码，以提高编码效率。反之，在对背景复杂，细节较多的视频进行编码时，编码器倾向于使用深度大的编码单元对图像帧进行处理，小尺寸的编码单元能很好地处理图像局部的细节，避免图像质量显著下降，减少失真。从图 3.1 编码单元划分实例中可以看出，图像中简单、平滑的区域，其编码深度一般为 0 或 1；复杂、纹理丰富的区域，其编码深度一般为 2 或 3。

图 3.1 编码单元划分实例

此外，本文对量化参数 QP 和编码单元划分深度进行了统计分析。通过算法 1 获取一帧图像编码单元深度信息，具体流程可分为两个步骤：

（1）为了直观地观察编码单元划分情况，把 Z 字扫描（Z scan）的顺序索引按照光栅扫描（Raster Scan）的扫描顺序重新扫描一遍。H.265/HEVC 对像素块的扫描方式有两

种：Raster Scan 和 Z scan。Raster 扫描是从上到下，从左到右进行扫描，是最直观也是最容易理解的方式。图 3.2 形象地说明了 Z 字扫描的方式，从左上角按照 Z 字形扫描到右下角。编码单元是递归划分的，为了方便处理，HM 中使用了 Z 字扫描。在 HM 中深度信息是通过 4×4 的矩阵进行存储，所以一个 64×64 的编码树单元深度信息在 HM 中需要用 16×16 的矩阵表示。

（2）分别统计深度为 0、1、2、3 的编码单元个数并打印。根据步骤（1）中的扫描顺序在 xCompressCU()函数中获取深度信息，定义四个全局变量 iCount0、iCount1、iCount2 和 iCount3 统计深度信息。

算法 1 编码树单元深度信息统计算法

```
1.输入:视频和编码配置信息
2.输出:编码树单元深度图和深度 0,1,2,3 编码单元个数
3.定义全局变量iCount0, iCount1, iCount2, iCount3
4.for i < TotalNumberPart do
5.   depth = getDepth(RasterToZscan[i])
6.   if depth = j j∈ [0,3] do
7.      iCountj + +
8.   end if
9.end for
```

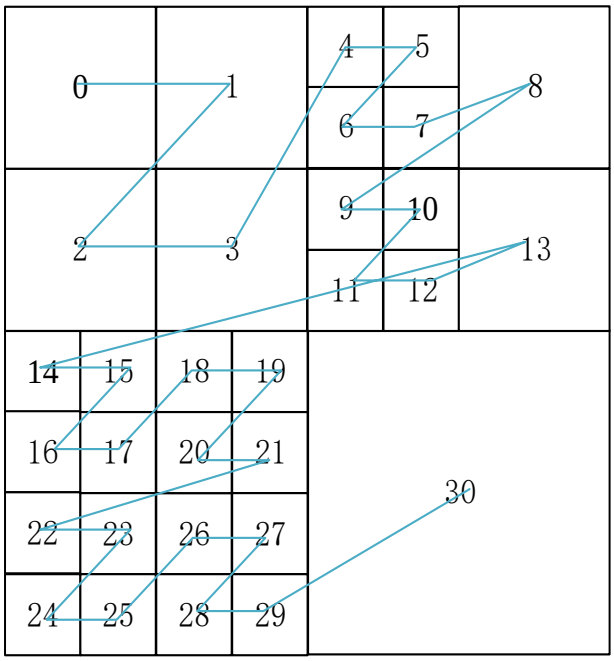


图 3.2 Z 字扫描

算法 1 在 H.265/HEVC 官方编解码软件 HM16.25 上实现，采用全帧内编码，量化参数设置为 22、27、32、37，配置文件为 encoder yuv source.cfg 和 encoder intra main.cfg，所用的测试序列是通用测试条件规定的序列 Traffic、BQTerrace、PartyScene 和 FourPeople。

实验结果如表 3.1 所示，其中，Sequence 代表视频序列名称。观察表 3.1 可知，随

着 QP 的增大，四个序列深度为 0 的编码单元占比都在增大，其中 FourPeople 增幅最大，由 2.1%增大到 11.2%；PartyScene 增幅最小，仅增加了 0.4%，这说明视频图像内容影响了编码单元的划分，如图 3.3 所示，相比于其他视频，PartyScene 背景复杂，细节较多，编码器倾向于使用大深度的编码单元进行编码；四个序列深度为 1 的编码单元占比逐渐增大，其中 Traffic 增幅最大，由 19.1%增长到 34.4%；编码单元深度为 3 时，Traffic、BQTerrace 和 PartyScene 占比在增大，但 FourPeople 占比值在上下波动；四个序列深度为 3 的编码单元占比显著下降，最大下降值为 34.7%。

表 3.2 统计了不同量化参数下 4 种深度的编码单元平均占比。由结果可知，随着量化参数的增大，深度为 0、1、2 的编码单元占比均呈上升状态，深度为 0 的编码单元占比从 3.14%上升到 9.05%，而深度为 3 的编码单元占比显著降低，从 60.50%下降到了 35.95%。通过统计分析可知，在 H.265/HEVC 中，量化参数会对编码单元的划分产生一定的影响。量化参数较小时，编码器倾向于使用深度大、尺寸小的编码单元，量化参数较大时，编码器倾向于使用深度小、尺寸大的编码单元。

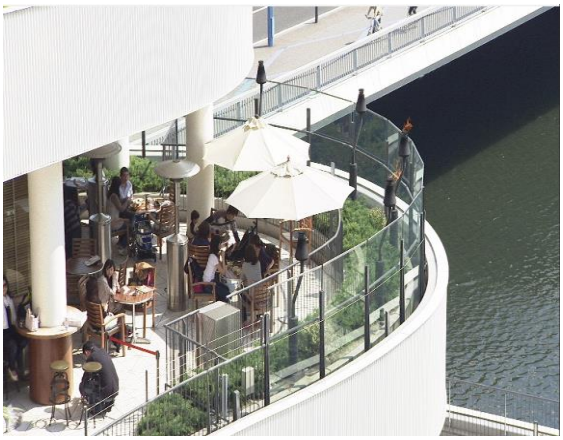
表 3.1 不同 QP 下 4 种深度的编码单元占比

Sequence		QP=22(%)	QP=27(%)	QP=32(%)	QP=37(%)
Traffic (2560×1600)	深度为 0	1.8	3.0	6.4	9.3
	深度为 1	19.1	24.3	28.1	34.4
	深度为 2	31.4	35.6	36.4	36.7
	深度为 3	47.7	37.1	29.1	19.6
BQTerrace (1920×1080)	深度为 0	8.6	13.3	14.3	15.7
	深度为 1	12.6	17.4	21.4	24.3
	深度为 2	11.2	19.1	23.6	27.1
	深度为 3	67.6	50.2	40.7	32.9
PartyScene (832×480)	深度为 0	0.1	0.2	0.4	0.5
	深度为 1	6.5	6.7	7.5	8.4
	深度为 2	5.6	8.8	11.5	21.2
	深度为 3	87.8	84.3	80.6	69.9
FourPeople (1280×720)	深度为 0	2.1	4.2	7.9	11.2
	深度为 1	23.4	28.3	30.4	32.7
	深度为 2	35.6	33.1	32.7	34.7
	深度为 3	38.9	34.4	29.0	21.4

表 3.2 不同 QP 下 4 种深度的编码单元平均占比

	深度为 0	深度为 1	深度为 2	深度为 3
QP=22(%)	3.14	15.41	20.95	60.50
QP=27(%)	5.13	19.23	24.15	51.49
QP=32(%)	7.15	21.85	26.15	44.85
QP=37(%)	9.05	25.07	29.93	35.95

(a) Traffic



(b) BQTerrace

(c) PartyScene

(d) FourPeople

图 3.3 四种视频序列

通过以上分析可知，编码单元快速划分技术的中心思想是：编码单元划分与视频图像内容和量化参数存在一定关系。但传统的帧内编码单元划分技术存在两点不足：（1）当编码块纹理复杂时，简单的拟合算法无法准确地描述编码单元的大小与视频图像内容和量化参数的关系；（2）划分决策是人为拟定的，且比较固定，而视频内容丰富，场景多样，导致该方法通用性差。

为了解决以上问题，本章提出基于回归思想的神经网络帧内编码单元快速划分算法，搭建合适的神经网络，很好地拟合编码单元的大小与视频图像内容和量化参数的关系，同时训练大量数据，提高网络的泛化性。

3.2 编码单元划分回归网络

目前，支持帧内编码单元快速划分的卷积神经网络存在层数太浅，训练集数据少的问题，可能出现拟合不足的现象。针对以上问题，同时考虑到加深网络带来的网络退化问题[35]，提出一种帧内编码单元划分回归网络 CuprNet（Coding Unit Partition Regression Network），以深度残差网络 ResNet（Deep Residual Network）为基础，优化 ResNet 网络全连接层与输出相匹配，加深网络层数、加大卷积核个数，提高算法性能，同时创建大型数据库供卷积神经网络学习，训练神经网络学习更多的特征。

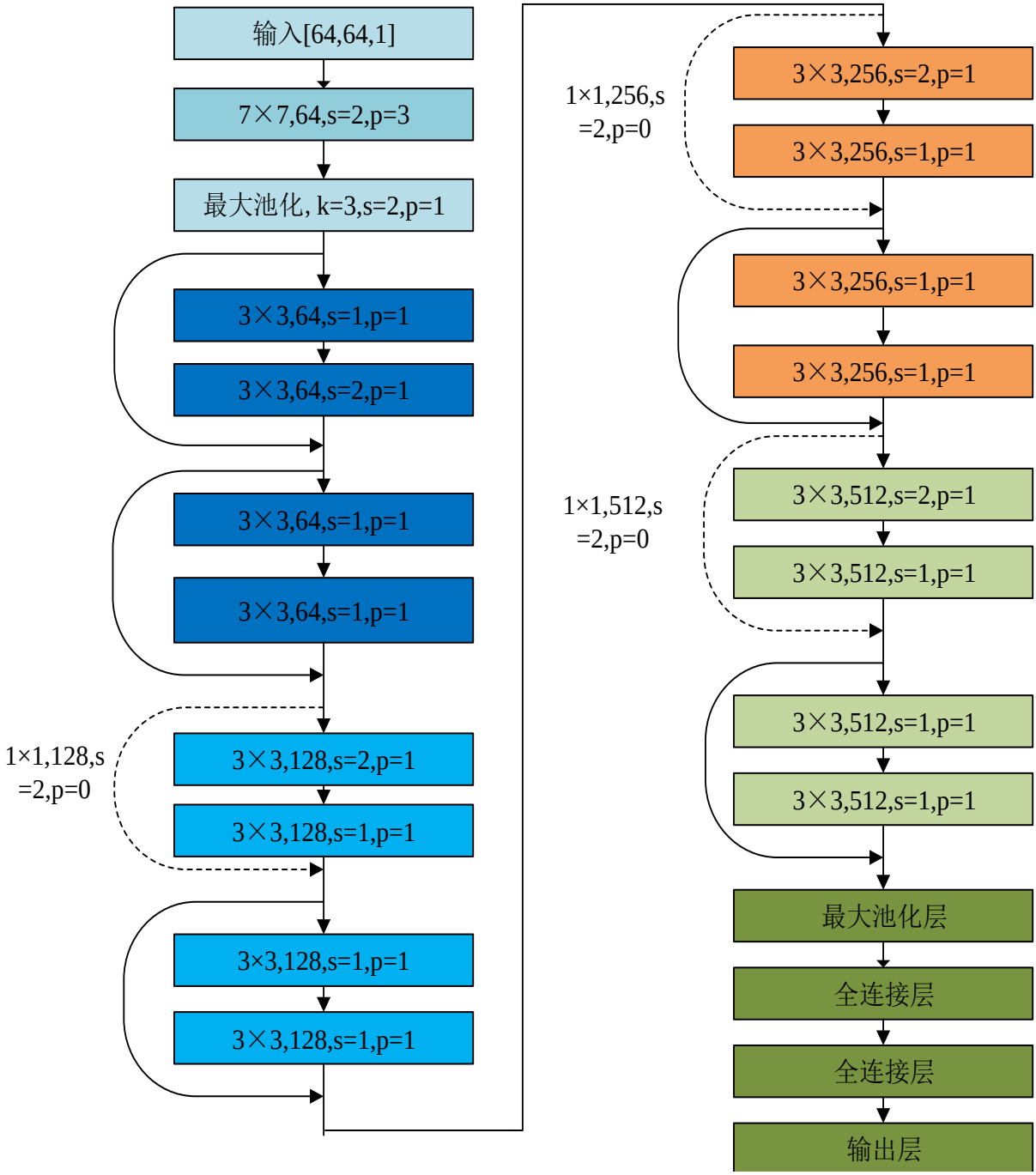


图 3.4 CuprNet 网络结构图

CuprNet 网络包括输入层、17 个卷积层、最大池化层、2 个全连接层和输出层。图 3.4 是 CuprNet 网络结构图，其中，s 代表步长，p 代表填充，1×1 和 3×3 都是卷积核，64、128、256、512 是卷积核个数。CuprNet 网络以 64×64 的编码树单元亮度分量作为输入，4×4 的编码树单元深度图作为输出。具体流程为：首先，构建包含视频亮度信息和编码树单元划分结构信息的预处理数据集，然后通过卷积层对预处理数据进行卷积操作，学习数据特征，卷积层使用线性修正激活函数，然后将数据特征输入最大池化层，以加快计算速度和防止过拟合，最后采用 3 层全连接层，包括 2 层隐藏层和 1 个输出层，在全连接层加入量化参数一起训练，最终输出编码树单元深度图。编码树单元深度图能够直观地表示图像划分结构。下面就对 CuprNet 网络的各个部分进行详细介绍。

（1）输入层和输出层

数据输入层对原始的视频图像数据进行去均值、归一化等预处理操作[38]。对图像数据进行预处理操作使数据标准化，方便后面的卷积层、池化层和全连接层对数据进行进一步处理，同时去除噪声等影响数据精度、准确度的因素，减少数据运算量，加速收敛。去均值化处理具体操作就是将视频每一帧特征值减去视频全部帧的特征均值，将原来在标准坐标系下的各个向量构成的矩阵变为以各向量的平均值为原点构建的坐标系。

CuprNet 的中心思想是拟合地编码单元的大小与视频图像内容的关系。本章采用视频图像像素表示视频内容，输入的编码树单元信息是从 YUV 格式的原始图像或序列中提取的，在 CuprNet 中只使用 Y 通道，因为该通道包含了最多的视觉信息[39]。CuprNet 网络对输入的亮度信息和量化参数进行归一化操作，计算公式如下：

$$X^{*}=\frac{X-X_{\min}}{X_{\max}-X_{\min}}$$
$$Q^{*}=\frac{Q-Q_{\min}}{Q_{\max}-Q_{\min}} \tag{3.1}$$

其中，X 为像素原始值，X_{min} 为像素最小值，X_{max} 为像素最大值，X^{*}为经过归一化处理后的值，Q 是量化参数原始值，Q_{min} 是量化参数最小值，Q_{max} 是量化参数最大值，Q^{*}为经过归一化处理后的值。

输出层直接获取编码树单元的划分深度图。由 2.2 节可知，深度 0 分别代表 64×64 大小的编码单元，深度 1 代表 32×32 大小的编码单元，深度 2 代表 16×16 大小的编码单元，深度 3 代表 8×8 大小的编码单元。64×64 的编码单元可划分为 64 个 8×8 的编码单元，即可用 8×8 的矩阵来表示一个编码树单元的深度信息，矩阵中的值为 0、1、2、3。根据四叉树划分原理，结合图 3.5 分析可知，使用 4×4 的矩阵就可以表示整个编码树单元的深度信息。

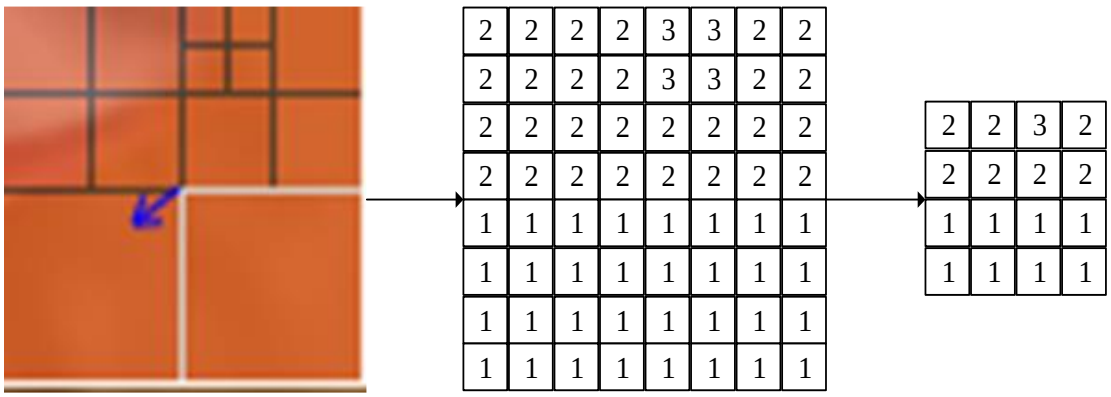


图 3.5 编码树单元深度信息

（2）特征提取模块

卷积层是卷积神经网络组成结构中最重要的一环，值得注意的是，深度神经网络中卷积的计算与传统的卷积计算公式不同，图 3.6 说明了神经网络卷积计算方式：

$$Y_{11}=(X_{11}W_{11}+X_{12}W_{12}+X_{21}W_{21}+X_{22}W_{22}) \tag{3.2}$$

在神经网络卷积层中，首先将输入图片与卷积核大小相同区域与卷积核中的权重按照图 3.6 方式进行计算，并将计算结果记录到输出块；然后按照规定的步长滑动卷积核，再次进行图 3.6 的操作，直至完成整张图片的计算。

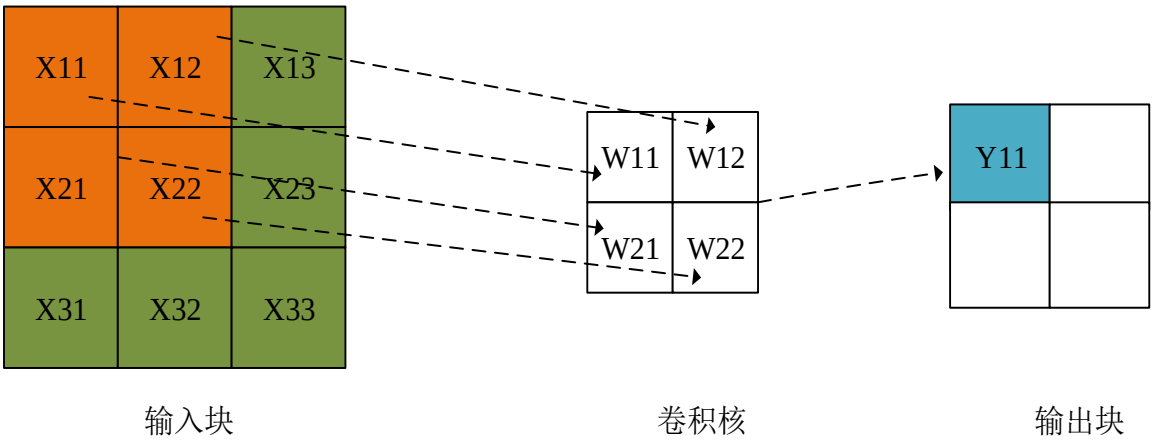


图 3.6 卷积运算

经过卷积运算后的输出与输入图片的尺寸不同，3 个超参数控制着输出的尺寸：深度，步长和零填充[40]。输出数据的深度与所用的卷积核数目相匹配，不同的卷积核在输入数据中所学到的特征是不一样的。滑动图片进行卷积运算，当步长为 n 时，每次移动 n 个像素。滑动至图片边缘时，若图片剩余区域与卷积核大小不匹配，需要用 0 在边缘处进行填充，称为零填充。卷积后输出尺寸的计算公式为：

$$H=\frac{N+2P-F}{S}+1 \tag{3.3}$$

其中，H 是输出尺寸，N 是输入图片的尺寸，F 是卷积核的大小，P 为填充值，S 是卷积步长。

在神经网络训练中，卷积层使用批量归一化数据进行批处理[41]，利用式（3.4）计算

得到的均值和方差来归一化输入，使得输出具有均值为 0，标准差为 1，从而缓解梯度消失，加速模型的收敛。

$$\begin{aligned}\mu_B &= \frac{1}{m}\sum_{i=1}^{m} x_i \\ \sigma_B^2 &= \frac{1}{m}\sum_{i=1}^{m}(x_i-\mu_B)^2 \\ \hat{x}_i &= \frac{x_i-\mu_B}{\sqrt{\sigma_B^2+\varepsilon}} \\ y_i &= \gamma\hat{x}_i+\beta\end{aligned} \tag{3.4}$$

式中，m 是批量数据的个数，μ_B 是批数据的平均值，σ_B 是方差，x_i 是第 i 个样本值，$\hat{x}_i$ 是标准化后的值，然后通过可学习的参数 γ 和 β 对 $\hat{x}_i$ 进行调整得到 y_i。

CuprNet 网络的 17 个卷积层构成特征提取模块，卷积层通过卷积操作对当前输入的 Y 分量进行特征提取，在卷积过程中得到特征 $f_1(Y), f_2(Y), \cdots f_{17}(Y)$，计算公式为：

$$\begin{cases} f_1(Y) = \delta(W_1 * Y) \\ f_2(Y) = \delta(W_2 * f_1(Y)) \\ \vdots \\ f_{17}(Y) = \delta(W_{17} * f_{16}(Y)) \end{cases} \tag{3.5}$$

其中，δ 表示 ReLU 激活函数，$W_1, W_2, \cdots W_{17}$ 表示卷积核，*表示卷积操作。

（3）池化层

池化层对从特征提取模块得到的特征图进行下采样操作，减小特征图的大小，以降低计算复杂度。数据经过池化操作后依旧保留着最重要的特征，去掉的只是无关紧要的信息，称之为特征不变性[42]。在特征不变的前提下，池化层减少了冗余信息，降低网络训练复杂度，同时也在一定程度上防止过拟合。池化层主要使用两种池化方法：最大池化（Max Pooling）和平均池化（Average Pooling）。最大池化方法如图 3.7，对于每个 n×n 的区域选出最大的数作为输出矩阵对应位置的值，平均池化对于每个 n×n 的窗口计算窗口平均值作为输出矩阵对应位置的值。CuprNet 在卷积层之后，接入了 2×2 的最大池化层，通过消除模糊的局部数据来减少节点的数量。

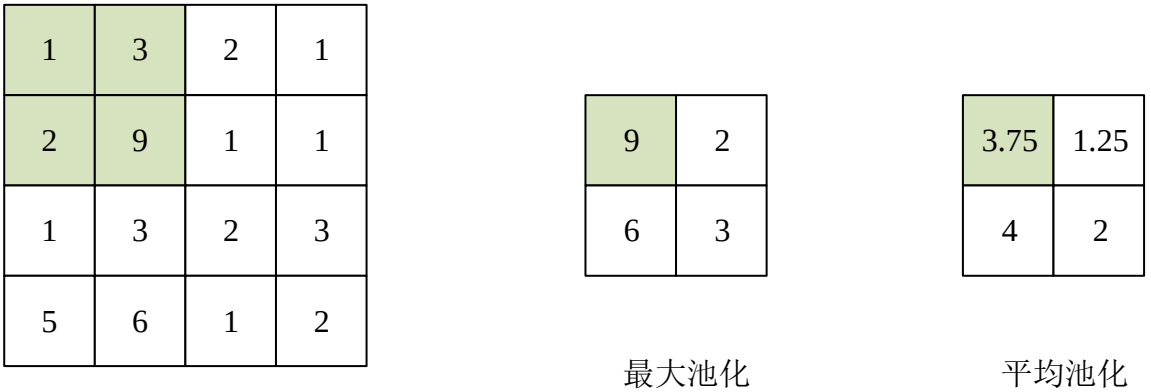


图 3.7 最大池化和平均池化

（4）全连接层

全连接层将池化层得到的 n 维矩阵转化为一维向量，将学到的特征高度提纯，减少了特征位置对网络训练结果的影响[43]。CuprNet 在卷积层使用批量归一化，在全连接层使用随机失活（Dropout），随机失活在神经网络在各个层次上的设置节点保留概率，减少节点之间的依赖，缓解了神经网络过拟合的现象。需要注意的是，Dropout 在测试阶段不需要使用，因为如果在测试阶段使用 Dropout 可能会导致预测值产生随机变化。

（5）网络测试和训练

在进行数据处理前，要建立一个合适的数据库，来训练和测试神经网络[44]。本文在 3.1 节调查和分析了影响编码单元划分的因素，最终得出编码单元尺寸与视频图像内容和量化参数有关。根据上述结果，对数据集进行构建。

首先，构建包含具有多个分辨率的视频序列集。随后，通过 H.265/HEVC 最新官方编解码器 HM16.25 对视频序列集进行编码，采用四个量化参数(QPs){22, 27, 32, 37}进行编码[1]，通过算法 2 得到编码后的编码树单元划分信息。算法 2 遍历了一帧图像中的所有编码树单元，为了与图像的像素点位置相匹配，对每个编码树单元按照光栅扫描的顺序获取深度信息，将获得的深度信息和量化参数写入文件中，重复以上操作，直到所有帧都已被编码。这样就获得了包含视频深度信息和量化参数的文件。随后读入视频的亮度信息，将亮度信息、深度信息和量化参数进一步整合获取预处理数据集。

算法 2 获取视频深度信息算法

```
1.输入:视频和编码配置信息
2.输出:视频深度信息和量化参数
3.frames = ToBeEnc()//获取视频待编码帧数
4.for i < frames do
5.   for x < 4 do
6.      for y < 4 do
7.         piCUDepthList[] = getDepth(g_auiRasterToZscan[]);//获取图像对应位置的深度
8.         qp[] = getQP()
8      end for
9.   end for
10.end for
```

神经网络的学习包括两个阶段：训练与测试。训练是包括前向传播和通过反向传播来调整网络权值两个步骤。在测试阶段，利用中间训练的网络及测试数据进行预测，根据预测的准确度及损失函数得到网络的性能。在训练阶段进行测试，能够对学习过程进行检验，从而避免在网络中出现过拟合现象。为了评价 CuprNet 网络的性能，本文使用均方误差函数作为网络的损失函数。

3.3 基于 CuprNet 网络的编码单元划分

基于CuprNet网络的编码单元划分算法框架如图3.8所示，主要包括视频输入模块、编码单元深度图模块、编码单元计算模块。首先将视频和编码配置信息输入到编码器中，然后根据网络模型和模型训练得到的参数获得该视频的编码树单元深度图，编码器根据得到的深度图计算和处理每个编码单元，最后存储编码单元编码信息。下面就算法的各个模块进行介绍。

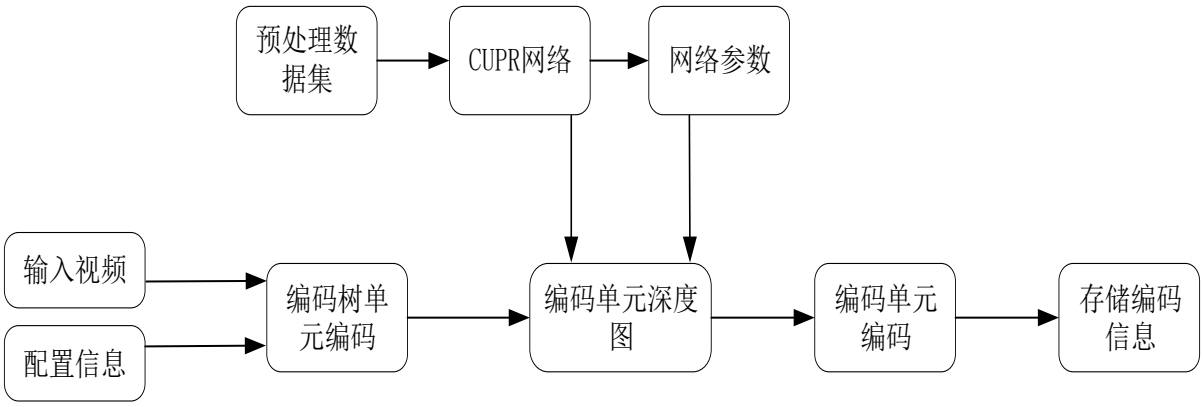


图 3.8 CuprNet 网络的编码单元划分算法

输入模块包括输入视频和编码配置信息两部分。编码器读取输入视频图像帧，将每一帧划分为若干个互不重叠的编码树单元，同时读取视频长度、宽度、量化参数、待编码帧数等编码配置信息，通过视频的长度和宽度可计算出一帧图像中有多少个待处理的编码树单元，量化参数作为外部特性加载到全连接层中，待编码帧数规定了通过 CuprNet 网络获取编码单元深度图的帧数。

编码单元深度图预测模块包括预处理数据集、全连接层优化 ResNet18 网络、网络参数、编码单元深度图四部分。使用深度图表示编码树单元划分结构，利用全连接层优化的神经网络模型对预处理数据集进行训练，得到合适的网络参数，将网络和参数应用到 HM-16.25 中，根据网络模型和参数获得视频实例编码树单元深度图。

编码单元计算模块如图 3.9 所示，编码器根据编码树单元深度图决定是否进行下一级编码单元划分，深度 0、1、2，分别对应 64×64、32×32、16×16 块的划分标志，确定编码单元划分后，对编码单元进行帧内最优预测模式决定和变换单元划分的计算。编码器在每个编码单元中递归地搜索最佳分区时，将当前深度与来自神经网络的预测深度进行比较。如果当前深度小于预测深度，那么它将继续搜索下一个深度。如果当前深度等于预测深度，那么它只做当前深度的计算，并停止搜索进一步的深度。如果当前深度大于预测深度，那么它不仅跳过当前深度的计算，还停止搜索进一步的深度。这确保了只有在当前深度是神经网络模型预测的深度时才会进行计算，减少了不必要的率失真代价的计算，减少运算复杂度。需要注意的是，网络输出层是一个 4×4 的矩阵，而 HM 中 64×64 的编码树单元深度信息用 16×16 的矩阵存储，在进行深度比较前，增加一个预处理模块，

将 4×4 的矩阵恢复至 16×16 的矩阵。

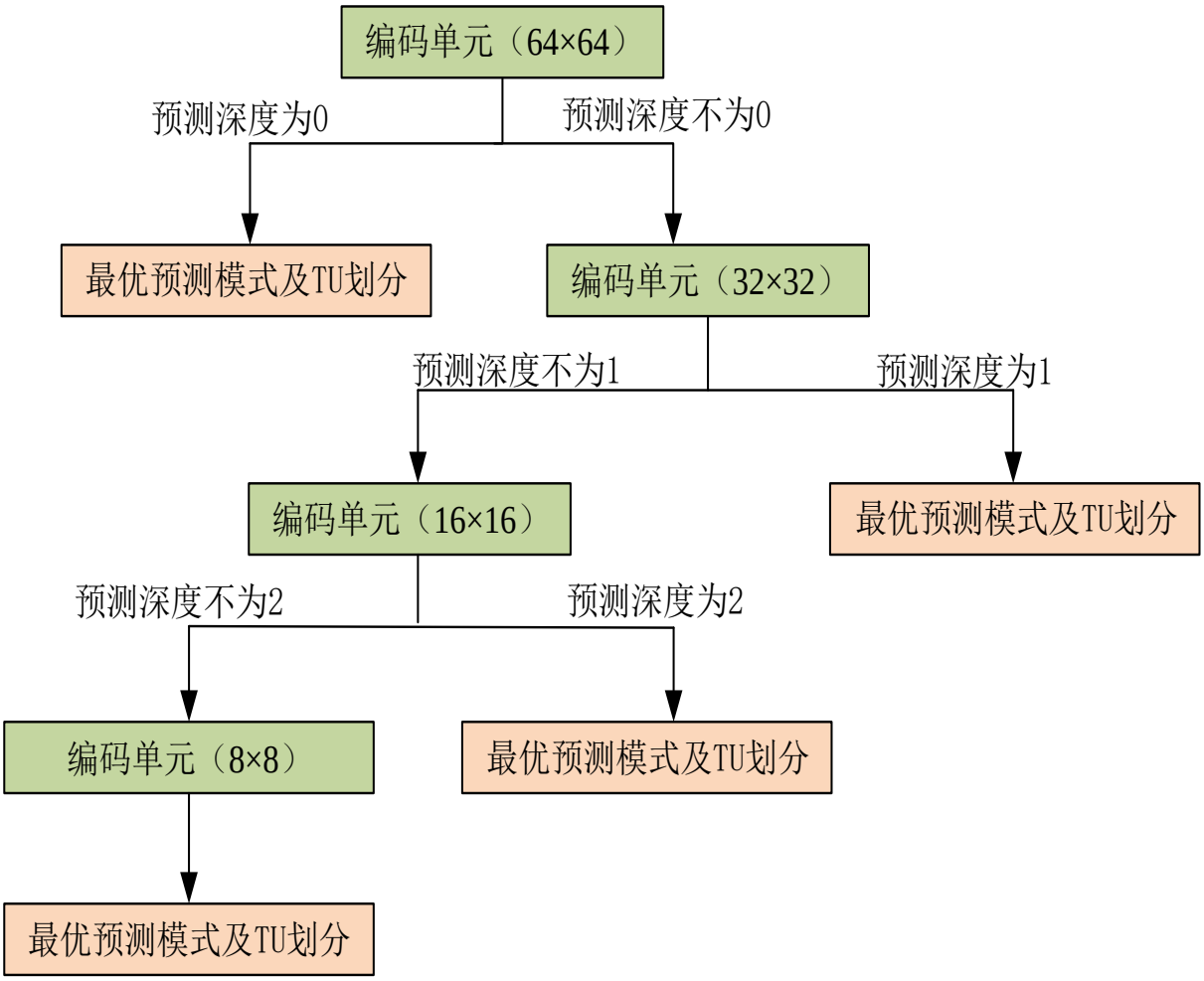


图 3.9 基于 CuprNet 网络的编码单元划分

3.4 性能评估

3.4.1 实验配置

（1）网络模型训练与嵌入

网络采用均方误差作为损失函数，在 TensorFlow 平台上借助 PyCharm 完成网络的训练。本文采用 Visual Studio 作为 CuprNet 网络训练得到的网络参数和 HM 软件测试平台的连接方。在 TensorFlow 平台将得到的网络参数存储到文件中，然后，在 Visual Studio 软件测试平台中修改 HM 代码，加载文件中的网络模型参数和网络模型，完成 CuprNet 网络在 HM 编码单元划分函数 xCompressCU()中的嵌入操作。

（2）编码过程中的配置

算法在 H.265/HEVC 官方编解码软件 HM16.25 上进行，开展了全帧内编码模型的测试，量化参数设置为 22、27、32、37，配置文件为 encoder yuv source.cfg 和 encoder intra main.cfg，所用的测试序列是通用测试条件规定的序列 ClassA、ClassB、ClassC 和 ClassE。

3.4.2 评价指标

为了评估所提算法的性能，采用 BD-psnr（Bjøntegaard Delta peak signai-to-noise rate）、BD-rate（Bjøntegaard Delta bit-rate）、编码节省时间 TS（Time Saver）等指标[45]。

BD-psnr 表示在给定的同等码率下，两种方法 PSNR-Y 差异，即优化后算法与原始算法相比视频客观质量的提高量[46]。BD-psnr 为正表示优化算法编码性能提高，反之，表示优化算法编码性能降低。BD-psnr 的计算如下式所示：

$$BD-psnr=\frac{\int_{R_1}^{R_2}(D_2-D_1)dr}{R_2-R_1} \tag{3.6}$$

其中，R 是码率，R_2 是两条曲线相同范围横坐标的最大值，R_1 是最小值，D_1、D_2 是三次多项式拟合曲线。

$$\begin{aligned} D_1&=a_1+b_1\times R+c_1\times R^2+d_1\times R^3\\ D_2&=a_2+b_2\times R+c_2\times R^2+d_2\times R^3 \end{aligned} \tag{3.7}$$

BD-rate 表示在同样客观质量下，优化算法与原始算法相比的码率增加量。BD-rate 为负表示优化后算法编码性能提高，反之，表示优化后算法编码性能下降。BD-rate 的计算如下式所示：

$$BD-rate=\frac{\int_{D_1}^{D_2}(R_2-R_1)dD}{D_2-D_1} \tag{3.8}$$

其中，D 是亮度 PSNR 值，D_2 是两条曲线相同范围横坐标的最大值，D_1 是最小值，R_1、R_2 分别是三次多项式拟合曲线。

$$\begin{aligned} R_1&=a_1+b_1\times D+c_1\times D^2+d_1\times D^3\\ R_2&=a_2+b_2\times D+c_2\times D^2+d_2\times D^3 \end{aligned} \tag{3.9}$$

用编码节省时间 TS 衡量编码复杂度降低程度，计算公式如下：

$$TS=\frac{T_{HM}-T_M}{T_{HM}}\times 100\% \tag{3.10}$$

其中，T_{HM} 是 HM16.25 从输入视频到所有帧编码结束的时间。T_M 是本文所提算法编码时间，是输入视频到输入视频编码单元深度图预测时间和网络模型预测结束到所有帧编码结束的时间之和。TS 越大代表优化后的算法编码复杂度越低。

3.4.3 结果分析

（1）编码性能分析

本章使用计算机仿真方法进行性能评估。实验结果如表 3.3 所示，其中 Class 代表序列类别，Sequence 是视频序列名称。

表 3.3 算法编码性能

Class	Sequence	BD-psnr(dB)	BD-rate(%)	TS(%)
A	Traffic	-0.4132	7.9543	81.42
(2560×1600)	PeopleOnStreet	-0.5370	9.7234	80.17
Class A	Average	-0.4751	8.8389	80.80
B	BQTerrace	-0.3726	6.2998	76.43
(1920×1080)	ParkScene	-0.2196	5.0186	81.44
Class B	Average	-0.2961	5.6592	78.94
C	PartyScene	-0.4644	6.0323	63.92
(832×480)	RaceHorses	-0.4213	6.5452	83.78
Class C	Average	-0.4429	6.2888	73.85
E	FourPeople	-0.6625	11.9887	85.21
(1280×720)	Johnny	-0.4399	9.9778	86.27
Class D	Average	-0.5512	10.9833	85.74
Average		-0.4413	7.9425	79.83

实验首先计算了采用传统方法和本章所提算法的时间节省率，通过时间节省率来比较两种算法的编码复杂度，具体的实验结果如表 3.3 所示。从表中可以看到，本章方法的编码时间明显小于传统算法的编码时间。与传统方法相比，本章所提算法平均编码复杂度降低了 79.83%，最高时间节省率高达 86.27%，最低时间节省率为 63.92%。实验证明本章提出的基于 ResNet18 的全连接层优化的帧内编码单元快速划分算法显著降低编码复杂度。

表 3.3 也给出了八个测试序列在编码后的算法性能，使用 BD-psnr 和 BD-rate 指标。可以看到，与传统算法相比，使用本章算法最高视频质量损失为 0.6625dB，最小损失为 0.2196dB，比特率最高增加 11.9887%，最少为 5.0186%，视频质量损失和比特率增加是相对应的。本章方法在八个标准测试序列上取得平均 0.4413dB 的 BD-psnr 损耗，在消耗相同码率的情况下，视频质量平均降低约 0.4413dB，等同于比特率平均增加 7.943%。实验证明本章提出的算法在牺牲一定视频质量的基础上，大幅度降低编码复杂度。

为了更直观地观察视频质量的变化，进行了以下实验：在全帧内模式，量化参数设置为 22 的条件下，对 FourPeople 序列进行编码，得到编码后的重构图像，并与传统的 HM16.25 编码后的图像进行对比。实验结果如图 3.10 所示，通过表 3.3 可知 FourPeople 在八个序列中视频质量损失最为严重，经基于 CuprNet 网络的编码单元划分算法编码后视频质量降低了 0.6625dB。对比图像可知，视频经过本章所提算法编码后的重构图像比 HM16.25 编码重构图像亮度稍低，但图像的主要特征没有丢失，也没有出现其它失真情况。

(a)FourPeople from Cupr

(b)FourPeople from HM

图 3.10 重构图像

（2）与其他方法的对比

将本章提出的算法与文献[19]、文献[1]进行对比，结果如表 3.4 所示，可以看出本章提出的算法时间节省率高于文献[19]18.41%，但是 BD-rate 高于参考文献 5.39%；提出的算法时间节省率高于文献[1]18.31%，但是 BD-rate 高于参考文献 5.99%。这说明算法以损失视频质量为代价，降低编码复杂度。通过对比可知还需对算法进行进一步优化，提升算法性能。

表 3.4 与参考文献对比

Class	Sequence	Ref19.		Ref1		Proposed	
		BD-rate	TS	BD-rate	TS	BD-rate	TS
A	Traffic	2.35	60.7	2.38	69.1	7.95	81.4
(2560×1600)	PeopleOnStreet	2.27	61.8	2.34	63.2	9.72	80.2
Class A	Average	2.31	61.3	2.36	66.2	8.84	80.8
B	BQTerrace	2.26	62.4	1.35	57.2	6.30	76.44
(1920×1080)	ParkScene	1.86	60.3	1.7	64.8	5.02	81.45
Class B	Average	2.06	61.4	1.53	61.0	5.66	78.94
C	PartyScene	2.19	51.1	0.49	41.4	6.03	63.92
(832×480)	RaceHorses	2.08	56.2	1.52	56.2	6.54	83.78
Class C	Average	2.14	56.9	1.01	48.8	6.29	73.85
E	FourPeople	3.05	60	2.71	65	11.99	85.22
(1280×720)	Johnny	4.42	72.2	3.16	75.2	9.98	86.28
Class E	Average	3.74	66.1	2.94	70.1	10.99	85.75
AVERAGE		2.56	61.43	1.96	61.53	7.95	79.84

（3）CuprNet 对不同 QP 编码单元划分的泛化能力

由 3.2 节知，该神经网络训练集数据来源于视频编码时的信息，在实验配置中，量化参数 QP 被设置为 22、27、32、37，为了探究该网络模型在使用其他量化参数进行编码时，是否具有良好的泛化能力，进行了本小节的实验测试。实验在 HM16.25 上进行，采用全帧内模式，量化参数设置为 17、30、42、47。实验结果如表 3.5 和表 3.6 所示，表中 Class 代表类别，Sequence 是视频序列名称。

表 3.5 不同 QP 下算法的 BD-psnr 和 BD-rate

Class	Sequence	BD-psnr(dB)	BD-rate(%)
A	Traffic	-0.3695	7.5252
(2560×1600)	PeopleOnStreet	-0.4429	7.9700
B	BQTerrace	-0.3957	7.2806
(1920×1080)	ParkScene	-0.2096	6.8348
C	PartyScene	-0.3758	7.7370
(832×480)	RaceHorses	-0.3897	8.6348
E	FourPeople	-0.4663	9.6812
(1280×720)	Johnny	-0.4244	9.9458
Average		-0.3842	8.2011

表 3.6 不同 QP 下算法的时间节省率

Class	Sequence	QP(%)	QP(%)	QP(%)
		17	30	42
A	Traffic	76.65	76.60	76.54
(2560×1600)	PeopleOnStreet	75.89	73.99	75.27
Class A	Average	76.27	75.30	75.91
B	BQTerrace	81.91	72.65	72.68
(1920×1080)	ParkScene	81.70	83.23	82.97
Class B	Average	81.81	77.94	77.83
C	PartyScene	77.62	66.71	63.29
(832×480)	RaceHorses	81.26	74.63	75.13
Class C	Average	79.44	70.67	69.21
E	FourPeople	83.54	79.43	81.26
(1280×720)	Johnny	84.31	81.38	78.17
Class D	Average	83.93	80.41	79.72
AVERAGE		80.36	76.08	75.66

表 3.6 是不同 QP 下算法的时间节省率，在 QP 为 17 时，八个视频序列的平均编码节省时间为 80.36%；在 QP 为 30 时，八个视频序列的平均编码节省时间为 76.08%；在 QP 为 42 时，八个视频序列的平均编码节省时间为 75.66%。与（1）中平均编码节省时间 79.83%相比，差别不大。表 3.5 是不同 QP 下算法的 BD-psnr 和 BD-rate，在 QP 为 17、30、42、47 时，平均 BD-rate 为 8.2011%，平均 BD-psnr 为-0.3842dB。与（1）中的实验结果对比，编码性能只有轻微的波动，说明提出的网络模型对不同 QP 帧内编码单元划分具有良好的泛化能力。

3.5 本章小结

本章实现了一种基于卷积神经网络的帧内编码单元快速划分方法。首先，针对传统编码单元划分算法存在的问题，构建了基于 ResNet18 的帧内编码单元快速划分网络，随后在视频编码树单元划分的过程中，利用构建的数据集进行训练，获得网络模型中的参数。然后，将训练好的网络嵌入到 H.265/HEVC 标准参考测试软件平台 HM16.25 中，优化传统的帧内编码单元划分算法。实验结果表明，与视频编解码标准中的编码单元划分方法相比，算法可以有效降低编码复杂度。该算法输出是编码深度图，可以直观地表示编码划分情况，算法时间节省率高，但基于回归模型的算法码率损失较大。

第4章 CupcNet 网络编码单元划分算法

本章提出了一种基于分类思想的帧内编码单元划分算法，将传统递归划分问题转变为分级的分类问题，并利用深度学习中的二分类模型来预测不同深度的编码单元是否进行划分，并将所提算法在 H.265/HEVC 官方编解码软件 HM 进行性能评估，验证结果表明该算法可以有效提高编码单元划分速度。

4.1 分类模型设计分析

卷积神经网络中，分类是指输入视频图像的特征向量 x，经过神经网络得到预测值 y，y 是一个标签值，它代表着物体的类别。卷积神经网络的分类任务，实际上是训练一个分类器。分类任务与回归任务都是根据学习到的数据表征，对输入数据进行运算，得到预测值，都是有监督的学习[47]。

回归模型和分类模型的区别在于：

1．得到的预测值不同。分类模型输出值是离散的，是标签值，用于判断事件所属的类别，是一种定性输出；回归模型输出值是连续的，是真实值，是一种定量输出。

2．目的不同。分类模型主要目的找到一个决策平面，对整个平面里面的数据进行分类；回归模型的主要目的是找到最优拟合，通过回归算法得到是一个最优拟合模型，这个模型最好接近数据集中每个点。

3．结果不同。分类的结果只有对错之分，而回归的结果则是以是否更接近真实值作为标准，而不是简单的判断对错。

分类模型可以分为三类：二分类（Binary Classification）、多分类（Multi-Class Classification0 和多标签分类（Multi-Label Classification）。二分类的定义为：通过训练集{(x1,y1),(x2,y2),...,(xn,yn)}进行学习，建立一个从输入空间 X 到输出空间 Y 的映射 f:X-Y，以便更好地理解和预测复杂的数据。其中，Y={-1,1}或{0,1}。图 4.1 是常用的二分类算法。

传统二分类算法	思想
LR逻辑回归模型	概率划分
SVM支持向量机	空间划分
K近邻算法	距离划分
决策树	信息量划分
朴素贝叶斯	条件概率公式

图 4.1 二分类算法

多分类学习是一种从输入空间 x 到输出空间 y 的映射，它通过训练集{(x1,y1), (x2,y2),..., (xn,yn)}来实现，其中 Y 的个数大于 2。多标签分类是指根据分类规则，每个样本可以预测为一个或多个类别。

H.265/HEVC编码器按照四叉树算法递归划分编码单元，编码单元最大尺寸是64×64，最小尺寸是 8×8。除最小尺寸的编码单元外，每个尺寸的编码单元都要进行“是”与“否”的选择，以决定是否向下一级划分。因此，可分别对 64×64、32×32、16×16 大小的编码单元设计二分类器。利用三个分类模型分别作为编码单元 64×64 划分为 32×32、32×32 划分为 16×16 和 16×16 划分为 8×8 的判断依据，输出值为 0 代表停止划分，输出值为 1 代表继续向下划分。

4.2 编码单元划分分类网络

对于帧间模式，文献[48]提出了三层提前终止框架并使用联合 SVM 学习这些特征，但 SVM 算法对大规模训练样本难以实施。文献[37]使用了三个基于卷积神经网络的分类器来决定是否在每个深度级别进行分割，然而分割标志仅从当前块获得，而不是使用整个编码树单元信息。

针对上述问题，提出编码单元划分分类网络 CupcNet（Code Unit Partition Classification Network），创建大型数据库进行训练，以学习更多特征，在网络中使用小卷积核，获得更准确的预测结果[49]。

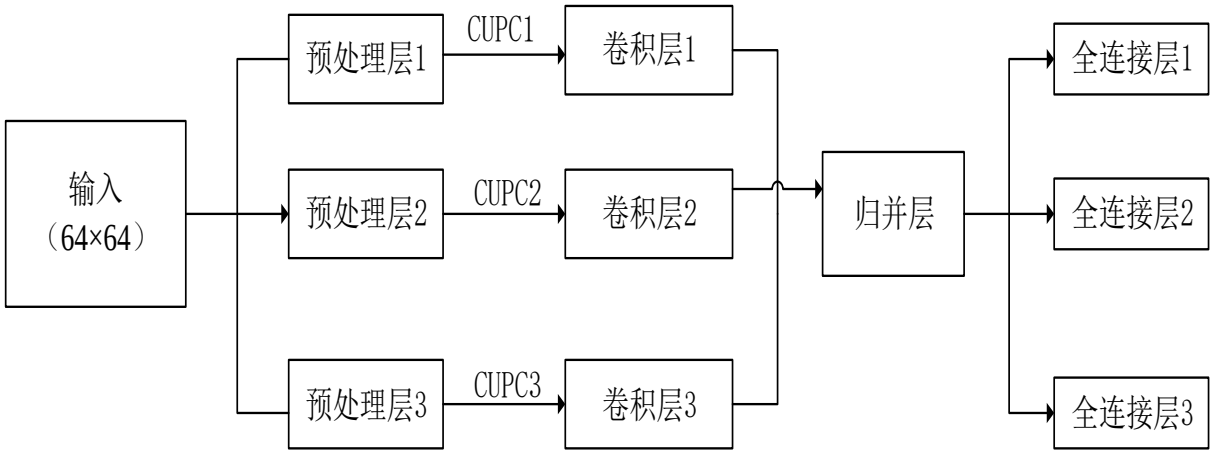


图 4.2 CupcNet 网络结构

CupcNet 网络结构如图 4.2 所示，由三个编码通道组成，分别为“深度=0”（64×64）、“深度=1”（32×32）和“深度=2”（16×16），从上到下依次对应。CupcNet 网络通过输入 64×64 的编号树单元亮度信息和量化参数，来预测当前深度编码单元的分割概率值[50]。

CupcNet 网络由 3 个预处理模块、3 组卷积层、1 个合并层以及 3 组全连接模块构成。预处理层经过三个平行分支对原始的编码树单元采用去均值化和向下采样的操作[51]。在每个分支处，对数据进行去均值化处理，加快神经网络的收敛速度。如图 4.3 所示，第二层和第三层对输入的编码树单元多一个预处理操作，将 64×64 大小的编码单元分别

下采样到 32×32、16×16 尺寸的编码单元，以满足分层训练的要求。

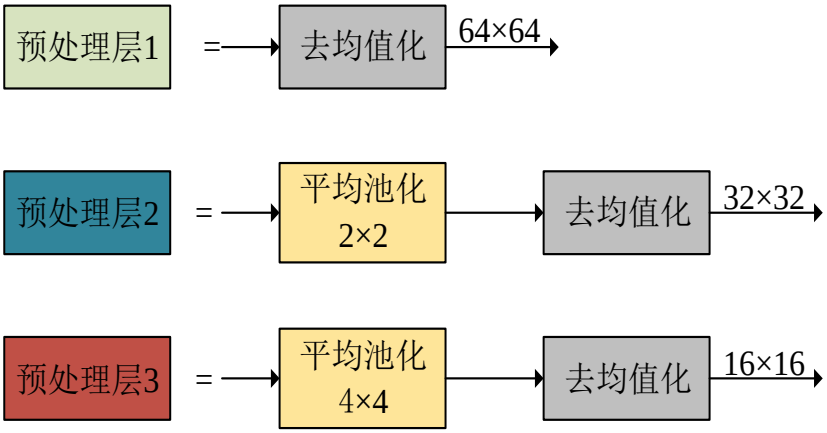


图 4.3 预处理层网络结构

预处理层后的不同尺寸编码单元分别接入不同的卷积层进行多尺度特征提取，卷积层具体网络结构如图 4.4 所示。编码树单元可以逐层完成特征提取，不同层采用不同的卷积核和卷积层数，以达到最佳的建模效果。但每个子卷积层均采用小卷积核，用多个小核卷积去替代大核卷积，可使网络获取更多的特征，提高网络准确度。同时，小卷积核减少了参数量，小卷积核的正则作用也能使网络性能提升，小卷积核使网络有更多的激活函数、更丰富的特征和更强的辨别能力[52]。为从预处理后的编码树单元中提取边缘特征，卷积层采用的激活函数是线性整流函数。卷积层 1 提取 64×64 编码单元的特征 $f_1(X_1), f_2(X_1), f_3(X_1), A_1(X_1)$：

$$\begin{cases} f_1(X_1) = \delta(W_1 * X_1) \\ f_2(X_1) = \delta(W_2 * f_1(X_1)) \\ f_3(X_1) = \delta(W_3 * f_2(X_1)) \\ A_1(X_1) = \delta(W_4 * f_3(X_1)) \end{cases} \tag{4.1}$$

卷积层 2 提取 32×32 编码单元的特征 $G_1(X_2), G_2(X_2), B_1(X_2)$：

$$\begin{cases} G_1(X_2) = \delta(I_1 * X_2) \\ G_2(X_2) = \delta(I_2 * G_1(X_2)) \\ B_1(X_2) = \delta(I_3 * G_2(X_2)) \end{cases} \tag{4.2}$$

卷积层 3 提取 16×16 编码单元的特征 $H_1(X_3), H_2(X_3), C_1(X_3)$：

$$\begin{cases} H_1(X_3) = \delta(L_1 * X_3) \\ H_2(X_3) = \delta(L_2 * H_1(X_3)) \\ C_1(X_3) = \delta(L_3 * H_2(X_3)) \end{cases} \tag{4.3}$$

其中，δ 是 ReLU 激活函数，W_1、W_2、W_3、W_4 是卷积层 1 的权重，I_1、I_2、I_3 是卷积层 2 的权重，L_1、L_2、L_3 是卷积层 3 的权重，*表示卷积操作。

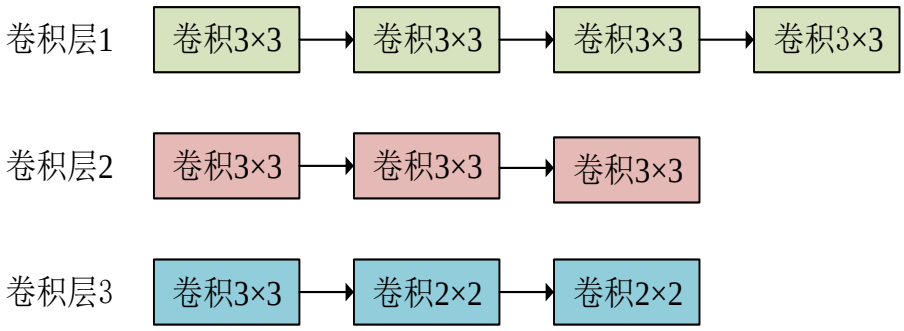


图 4.4 卷积层网络结构

归并层将第一层、第二层和第三层得到的特征归融合在一起，这样可以获得整个编码树单元的多个学习特征。整个编码树单元生成的特征都输入到三个全连接层模块中，而不是仅利用一个分支的编码树单元的特征，来预测当前尺寸的编码单元是否继续划分。将卷积层 1 的输出 $A_1(X_1)$，卷积层 2 的输出 $B_1(X_2)$ 和卷积层 3 的输出 $C_1(X_3)$ 进行数据拼接得到融合特征：

$$Z_1(X)=[A_1(X_1),B_1(X_2),C_1(X_3)] \tag{4.4}$$

归并层后接入三层平行的全连接模块，每层全连接模块包括两个全连接层和一个输出层。考虑到量化步长 QP 对编码单元划分有影响，在全连接层中，将 QP 当作一种外部特性，加入到特征向量中，以建立 QP 与编码单元划分的联系，从而可以更精确地预测划分结果，提高算法的性能。鉴于 CupcNet 网络的输出是二分类问题，即用 0 和 1 来表明编码单元能否被划分，所以，最后一层使用 Sigmoid 函数作为激活函数。全连接层模块的输入是特征和量化参数，经过卷积操作后得到输出，输出是概率值，该过程由下式给出：

$$\begin{cases} I_1(X)=[Z(X),QP] \\ F_1(X)=\delta(N_1*I_1(X)+b_1) \\ F_2(X)=\delta(N_2*F_1(X)+b_2) \\ O(X)=\sigma(N_3*F_2(X)+b_3) \end{cases} \tag{4.5}$$

式中，是 δ 是 ReLU 激活函数，σ 是 Sigmoid 激活函数，N_1、N_2、N_3 是权重，b_1、b_2、b_3 是偏移值，*表示卷积操作。

在训练 CupcNet 网络时，采用交叉熵损失函数作为网络的损失函数。在信息论中，熵用来衡量事物的不确定性。熵越大，事件的不确定性越高；熵越小，事件的不确定性越小。交叉熵损失函数（Cross-entropy loss）是用来评估当前训练得到的概率分布与真实分布的差异情况[53]，交叉熵的值越小，两个概率分布就越接近。二分类交叉熵损失函数定义为：

$$loss=-\frac{1}{N}\sum_{n=1}^{N}(l_n\times\log Y_n+(1-l_n)\times\log(1-Y_n)) \tag{4.6}$$

其中，l_n 表示第 n 个样本真实的标签值，Y_n 是第 n 个样本预测的概率值，N 是总样本个

数。

在训练网络前，要创建数据集。在 CupcNet 网络中，一个编码树单元对应的标签数是 21，其中，标签值为 0 或者 1，0 代表不继续划分，1 代表向下一级划分。编码树单元划分标签具体组成结构如表 4.1 所示，可用一维数组 a[i]表示，i 代表当前编码单元在 64×64 编码树单元的位置。其中 a[0]存储的是 64×64 编码单元划分信息，a[1]~a[4]存储的是 32×32 编码单元划分信息，a[5]~a[20]存储的是 16×16 编码单元划分信息。因此，需要将 3.2.2 节中获取的深度信息映射成表 4.1 的组成形式，随后读入视频的亮度信息，将亮度信息、标签信息和量化参数进一步整合，从而获取帧内编码单元快速划分分类模型的预处理数据集。

表 4.1 编码树单元划分标签的组成形式

深度	划分：0 不划分：1			
0	1			
1	0	1	0	1
2	0000	1010	0000	0010
3	不再继续向下划分			
组成形式	10101000010100000010			

4.3 基于 CupcNet 网络的编码单元划分

提出的基于 CupcNet 网络的帧内编码单元快速划分算法流程图如图 4.5 所示。首先，编码器读取视频和编码配置信息；其次，读取待编码 CTU，并在编码单元划分模块调用 CupcNet 网络；然后，利用 CupcNet 网络预测输入视频的编码单元划分结构；最后，获得编码树单元划分信息，编码器继续编码至编码结束。下面对各部分进行详细介绍。

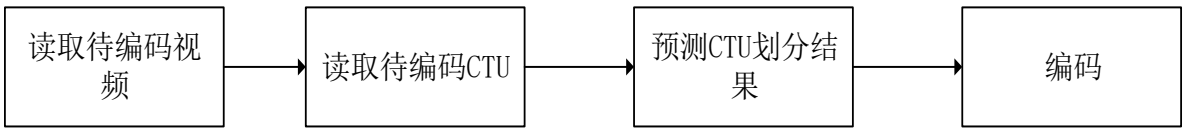


图 4.5 CupcNet 编码单元划分优化框架

读取待编码视频模块获取视频和编码配置信息。编码器读取输入视频图像帧，将每一帧划分为若干个互不重叠的编码树单元，同时读取视频长度、宽度、量化参数、待编码帧数等编码配置信息。

预测 CTU 划分结果模块输入为 CupcNet 网络、网络参数、图像亮度信息和量化参数，如算法 3 所示，具体流程可分为两个步骤：

1．HM-16.25 中，编码器逐帧处理视频序列，在编码单元划分模块调用 CupcNet 网络和训练得到的网络参数，输入图像亮度信息和量化参数得到当前帧编码单元的划分结果，将预测结果输入到文件中。

2．根据编码单元尺寸和位置，读取文件中对应的预测结果，根据结果决定是否向下

一级划分。HM通过嵌套调用xCompressCU()函数对编码单元进行划分，在xCompressCU()中定义变量 ifsub，当 ifsub=false 代表不继续划分，ifsub=true 代表继续划分，根据预测结果给 ifsub 赋值，可提前终止编码单元的划分，减少拉格朗日率失真函数的计算，降低编码复杂度。

算法 3 编码单元划分算法

```
1.输入:CupcNet 网络、网络参数、图像亮度信息和量化参数
2.输出:编码单元划分结果
3.frames=ToBeEnc()//获取视频待编码帧数
4.for i<frames do
5.   if size=8 do
6.      ifsub=false
7.   else index=i i=1,2,…20//根据 ctu 大小获取对应的表 4.1 数组中的下标
8.        split=cudepth[21*iCTUAddr+index]//获取对应位置预测结果
9.        if split<0.5 do
10.              ifsub=false
11.       else ifsub=true
12.       end if
13.   end if
14.end for
```

4.4 性能评估

4.4.1 实验配置

为了验证本章所提算法性能，需要先训练提出的 CupcNet 网络，训练集包含视频图像的亮度信息、量化参数和对应的标签信息。CupcNet 训练基于 CPU 环境使用 TensorFlow 架构，借助 PyCharm 完成模型的训练。模型训练过程中，原始编码单元尺寸为 64×64，经过通道 1、通道 2 和通道 3 预处理后的编码单元的大小为 64×64、32×32、16×16。学习率动态调节，随着所学次数的增多，学习率会逐渐减少，以避免固定学习率带来的收敛缓慢和最优解的问题[54]。

在 H.265/HEVC 测试平台 HM16.25 上开展了全帧内编码模型的测试，使用 Visual Studio 2017 编译软件，配置文件为 encoder yuv source.cfg 和 encoder intra main.cfg。使用了标准测试顺序，其技术参数也已经详细说明。

4.4.2 评价指标

评价 CupcNet 网络时采用了混淆矩阵（Confusion Matrix），也称为误差矩阵，n 行 n 列，是一组用于衡量机器学习模型的规范形式，可以用来衡量模型准确性和可靠性。在机器学习模型评价中，混淆矩阵是一个重要的指标，混淆矩阵常用于分类器和各种分类别的统计建模，比如分类树、逻辑回归、线性判别分析等[55]。

图 4.6 是二分类的混淆矩阵。混淆矩阵由如下两个维度构成：样本的实际标签和样本被模型预测出来的标签。其中列表示模型预测（Predict）值，行表示数据真实（True）值。在这两个维度下交织成四个可能的情况：真实值为 1，预测值也为 1，用字母 TP（True Positive）表示真正例数据；真实值为 1，预测值为 0，用字母 FN（False Negative）表示假反例数据；实际值和预期值都是 0，用字母 TN（True Negative）来表达真反例数据；真实值为 0，预测值为 1，用字母 FP（False Positive）表示假正例数据。True 和 False 分别代表预测结果的正确性，而 Positive 和 Negative 则表明预测结果是否具有正向或负向的倾向。

混淆矩阵		预测值	
		1	0
真实值	1	TP	FN
	0	FP	TN

图 4.6 二分类混淆矩阵

根据混淆矩阵，采用准确度作为网络性能评价指标，准确率是指模型预测正确的结果占总结果的比重，准确率定义为：

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.7}$$

4.4.3 结果分析

（1）CupcNet 网络性能分析

通过准确率来评价 CupcNet 卷积神经网络的性能，实验结果如图 4.7 所示，图 4.7 反映了 CupcNet 网络准确率的变化情况。其中，青色对应针对 64×64 尺寸的编码单元设计的网络（神经网络 1），红色对应针对 32×32 尺寸的编码单元设计的网络（神经网络 2），绿色对应针对 16×16 尺寸编码单元设计的网络（神经网络 3）。由图可知，神经网络 1 的准确率随着训练次数的增加逐步稳定在 0.8，神经网络 2 的准确率高于神经网络 3，且随着训练次数的增加趋于平稳，神经网络 3 的准确率最低，接近于 0.8，验证集和训练集准确率曲线趋势相似。综上所述，不同尺寸的编码单元的准确率均在 0.8 附近波动，CupcNet 网络能够有效学习特征，得到较好的预测效果。

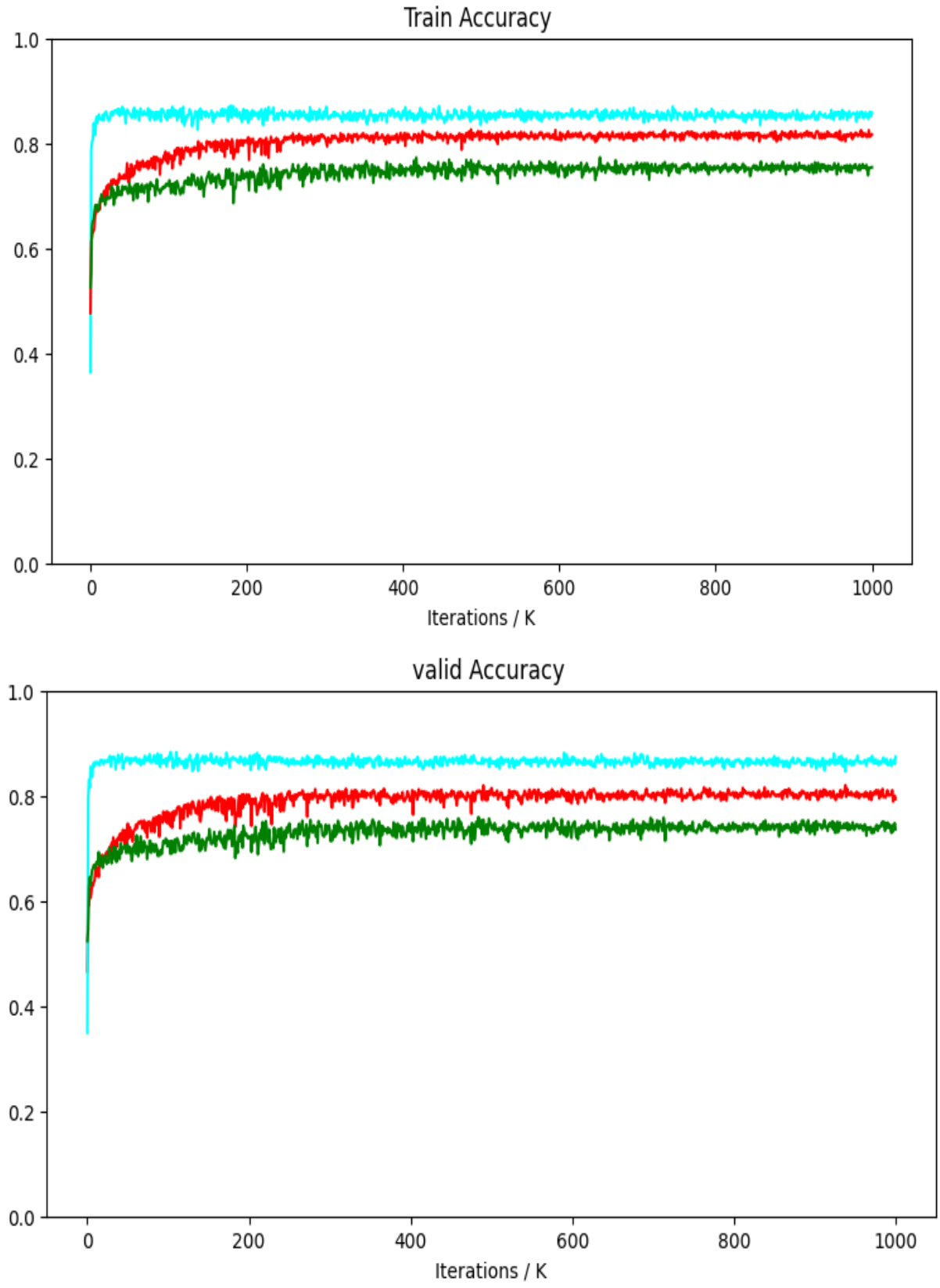


图 4.7 网络准确率

（2）与其他方法对比

实验首先计算了采用传统方法和本章所提算法的时间节省率，通过时间节省率来比较两种算法的编码复杂度，具体的实验结果如表 4.2 所示。从表中可以看到，与传统方

法相比，本章所提算法平均编码复杂度降低了 64.01%，最高时间节省率高达 75.36%。实验证明本章提出的算法能够减少 H.265/HEVC 编码器编码时间。

表 4.2 也给出了八个测试序列在编码后的 BD-psnr 和 BD-rate。可以看出，使用所提算法最高视频质量损失为 0.2485dB，最小损失为 0.0053dB，平均为 0.1508dB。比特率最高增加 5.2011%，最少为 0.1406%，平均为 2.9493%。综上可知，在可接受的视频损失范围内，本章提出的基于 CupcNet 网络的帧内编码单元快速划分算法显著降低编码复杂度。

表 4.2 与官方 HM16.25 对比结果

Class	Sequence	BD-PSNR(db)	BD-RATE(%)	TS(%)
A	Traffic	-0.1664	3.9354	65.09
(2560×1600)	PeopleOnStreet	-0.1597	2.7726	58.57
B	BQTerrace	-0.1356	2.1153	57.47
(1920×1080)	ParkScene	-0.1274	2.9789	70.91
C	PartyScene	-0.0053	0.1406	57.61
(832×480)	RaceHorses	-0.1369	2.1412	60.95
E	FourPeople	-0.2485	4.3089	66.08
(1280×720)	Johnny	-0.2266	5.2011	75.36
Average		-0.1508	2.9493	64.01

将本章提出的算法与文献[19]，文献[20]中所提的两种算法进行比较，结果如表 4.3 所示，可以看出本章提出的算法时间节省率高于文献[19]4.06%，高于文献[20]1.04%，BD-rate 小于文献[19]，大于文献[20]但差距不大，综合以上指标可得本文所提算法性能高于文献[19]。

表 4.3 与以往算法对比结果

Class	Sequence	Ref19		Ref20		Proposed	
		BD-rate	TS	BD-rate	TS	BD-rate	TS
A	Traffic	2.58	58.49	2.31	66.37	3.94	65.09
(2560×1600)	PeopleOnStreet	2.71	59.91	2.29	63.12	2.77	58.57
B	BQTerrace	2.68	61.06	1.48	63.26	2.12	57.47
(1920×1080)	ParkScene	1.90	59.58	2.06	58.91	2.98	70.91
C	PartyScene	2.79	54.71	1.43	61.02	0.14	57.61
(832×480)	RaceHorses	3.20	56.35	1.79	59.08	2.14	60.95
E	FourPeople	3.25	61.05	3.51	65.71	4.31	66.08
(1280×720)	Johnny	4.6	68.45	3.42	66.32	5.20	75.36
AVERAGE		2.96	59.95	2.29	62.97	2.95	64.01

（3）CupcNet 对于不同 QP 的编码单元划分泛化能力

由上文可知，训练数据集仅由 QP 为 22、27、32、37 编码后数据构成。为了探究该算法在对使用其他 QP 编码的图像进行帧内编码单元划分是否具有良好的泛化能力，进行了本小节的测试。实验在 HM16.25 上进行，采用全帧内模式，量化参数设置为 17、30、42、47。实验结果如表 4.4 和表 4.5 所示。

表 4.4 不同 QP 下的 BD-psnr 和 BD-rate

Class	Sequence	BD-PSNR(db)	BD-RATE(%)
A	Traffic	-0.2528	4.9935
(2560×1600)	PeopleOnStreet	-0.2702	5.0885
B	BQTerrace	-0.2297	4.9472
(1920×1080)	ParkScene	-0.1756	5.0989
C	PartyScene	-0.1014	3.3482
(832×480)	RaceHorses	-0.1779	4.5271
E	FourPeople	-0.3203	6.07876
(1280×720)	Johnny	-0.2373	5.3743
Average		-0.2206	4.9321

表 4.5 不同 QP 下编码节省时间

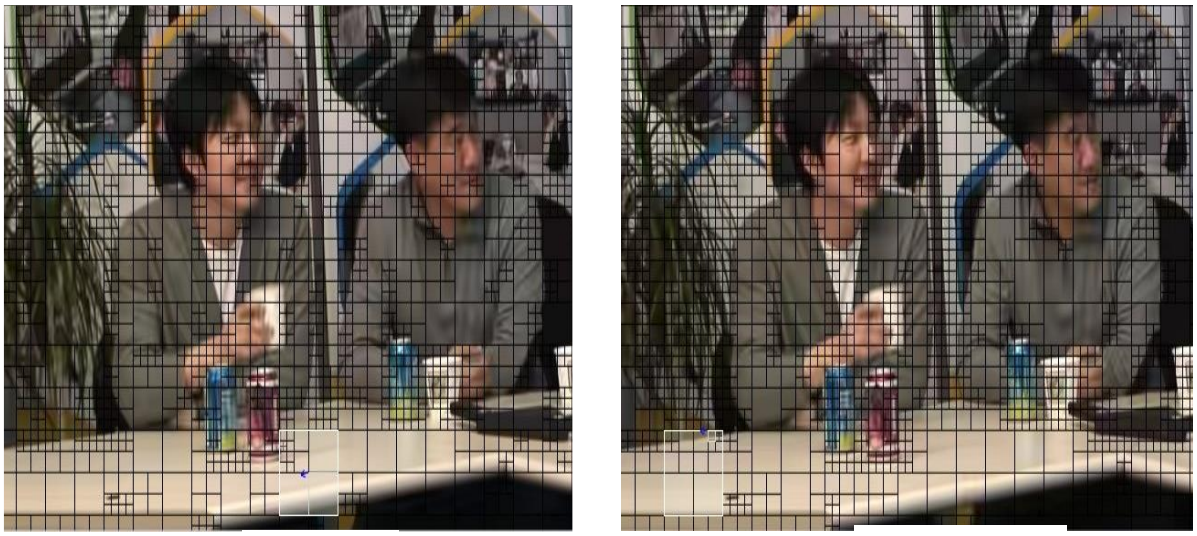
Class	Sequence	QP(%)	QP(%)	QP(%)
		17	30	42
A	Traffic	70.70	79.45	81.61
(2560×1600)	PeopleOnStreet	66.00	74.98	77.45
Class A	Average	68.35	77.22	79.53
B	BQTerrace	62.07	55.89	68.80
(1920×1080)	ParkScene	63.99	76.38	80.08
Class B	Average	63.03	66.14	74.44
C	PartyScene	50.77	51.74	53.49
(832×480)	RaceHorses	57.91	63.18	70.06
Class C	Average	54.34	57.46	61.78
E	FourPeople	66.11	74.57	78.15
(1280×720)	Johnny	74.68	81.25	83.82
Class E	Average	70.40	77.91	80.99
AVERAGE		64.03	69.68	74.19

表 4.5 是不同 QP 下算法的时间节省率，实验证明，在 QP 为 17、30、42 时，八个视频序列的平均编码复杂度降低了 64.03%、69.68%和 74.19%，与（2）中的实验结果并无太大差别。根据表 4.4 可知，BD-rate 为 4.9321%，视频质量损失为 0.2206dB，说明提出

的网络模型对于不同 QP 帧内编码单元快速划分预测具有较为良好的泛化能力。

（4）客观编码性能分析

为了更直观地观察视频质量损失情况和编码单元划分结构，在全帧内模式下，量化参数设置为 37 的情况下，对 BQTerrace 和 RaceHorses 序列进行编码，并且打印出编码过程中帧内编码单元划分情况。结果如图 4.8 所示，左侧图片是 BQTerrace 在 HM16.25 编码过程中帧内编码单元的划分情况，图 4.8 右侧图片是 BQTerrace 在提出的算法编码过程中帧内编码单元的划分情况，观察可知，编码单元划分结构相似，CupcNet 网络准确率较高。



(a)HM　　(b)CupcNet

图 4.8 编码单元划分情况

图 4.9 左侧图片是 RaceHorses 经过 HM16.25 编码后的重构图像，图 4.9 右侧图片是 RaceHorses 经过提出的算法编码后的重构图像。与表 4.4 的数据相对应，对于 RaceHorses 序列视频损失质量为 0.1129dB，观察编码重构后的图片，图像的主要特征没有丢失，也没有出现其它失真情况，算法具有良好的性能。

(a)HM　　(b)CupcNet

图 4.9 重构图像

4.5 本章小结

本章实现了一种基于分类思想的帧内编码单元快速划分算法。首先对视频图像内容、量化参数与帧内编码单元划分的关系进行定性分析，通过搭建卷积神经网络进行定量分析；提出了编码单元快速划分分类网络 CupcNet，将训练好的网络嵌入到 H.265/HEVC 标准参考测试软件平台 HM16.25 中，优化传统的帧内编码单元划分算法。实验结果表明，与视频编解码标准中的编码单元划分方法相比，所提算法可以有效降低编码复杂度。该算法码率损失较小，图像失真较小，但该算法时间节省率小于第 3 章提出的算法，且网络输出是 21 个标签值，而第 3 章回归网络的输出仅为 4×4 的矩阵。

第5章 全文总结与展望

5.1 全文总结

近些年来，随着 5G 时代的来临和人工智能的发展，将深度学习赋能于视频编解码，已成为研究趋势，由于帧内编码单元划分在 H.265/HEVC 官方编码器 HM 中占据 80%的时间，本文在深入研究编码单元划分技术和深度学习技术的基础上，进行了 CuprNet 网络编码单元划分算法和 CupcNet 网络编码单元划分算法两项工作。

（1）基于回归模型的 CuprNet 网络编码单元划分算法。首先，充分研究了 H.265/HEVC 中编码单元划分技术和在 HM 中具体实施步骤，分析得出决定编码单元尺寸的关键因素，即量化参数和视频图像纹理复杂度；其次，分析现有的利用深度学习的帧内编码单元快速划分算法，针对现有的神经网络层数浅，训练集数据量少的问题，提出全连接层优化的 CuprNet 网络，并构建了大型数据集；随后，将搭建的神经网络实际应用在 H.265/HEVC 编解码中，提出基于 CuprNet 网络的编码单元划分优化框架，主要包括视频输入模块、编码单元深度图模块、编码单元计算模块；最后，将提出的算法在 H.265/HEVC 编解码框架中进行性能评估验证，实验证明，与官方编码器相比，该算法在平均增加 7.943%的比特率的基础上平均降低了 79.83%的编码时间。

（2）基于分类模型的 CupcNet 网络编码单元划分算法。首先，从一个新的角度设计用于帧内编码单元模式快速模式抉择的卷积神经网络，即用分类的思想看待问题；针对帧内编码单元快速划分分类网络缺少对整个编码树单元信息使用的问题，在编码单元快速划分分类网络中加入用于特征融合的归并层，该神经网络包含三个子神经网络，分别用于训练 64×64、32×32、16×16 的编码单元，该网络充分学习了整个编码树单元的特征，同时，使用小卷积核代替大卷积核，增加了非线性激活函数提升模型的能力，减少了参数量；随后，提出基于 CupcNet 网络的编码单元划分优化框架，将提出的网络应用于实际编码器中；最后，通过实验验证了该算法对不同量化参数具有泛化性，实验结果表明，与官方编码器相比，该算法在平均增加 2.9493%比特率的基础上降低了 64.01%的编码时间。

5.2 工作展望

本文基于深度学习，对 H.265/HEVC 帧内编码单元快速划分技术进行了研究，降低了编码复杂度，但是本文的工作仍有待于深入探讨与完善。

（1）本文提出的算法均在计算机软件中进行仿真，缺少对硬件方面的分析，在以后的研究中，可以进一步对算法在硬件中实现进行详细的分析，针对性设计模块，以适用于实际的应用场景。

（2）深度学习不仅应用在帧内编码单元划分技术中，还可与视频编解码标准的其它模块相结合，比如预测模式选择、环路滤波、视频后处理、码率控制等模块，今后可对深度学习在 H.265/HEVC 其他技术中的应用进行研究，以优化 H.265/HEVC 编解码框架。

（3）本文从监督学习的角度提出帧内编码单元快速划分的算法，在今后的研究中，可以用无监督学习和强化学习的新角度解决帧内编码单元快速划分的问题。

参考文献

[1] Feng A, Gao C, Li L, et al. Cnn-based depth map prediction for fast block partitioning in hevc intra coding[C]. 2021 IEEE International Conference on Multimedia and Expo (ICME). IEEE, 2021: 1-6.

[2] Shlien S. Guide to MPEG-1 audio standard[J]. IEEE Transactions on Broadcasting, 1994, 40(4): 206-218.

[3] Tudor P N. MPEG-2 video compression[J]. Electronics & communication engineering journal, 1995, 7(6): 257-264.

[4] ITU-T. Video Codec for Audiovisual Services at p×64 Kbit/s[R]. Geneva, 1990.

[5] Rijkse K. H.263: Video coding for low-bit-rate communication[J]. IEEE Communications magazine, 1996, 34(12): 42-45.

[6] Marpe D, Wiegand T, Sullivan G J. The H.264/MPEG4 advanced video coding standard and its applications[J]. IEEE communications magazine, 2006, 44(8): 134-143.

[7] Wien M. High efficiency video coding[J]. Coding Tools and specification, 2015, 24.

[8] Bross B, Wang Y K, Ye Y, et al. Overview of the versatile video coding (VVC) standard and its applications[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2021, 31(10): 3736-3764.

[9] 高文，王强，马思伟．AVS 数字音视频编解码标准[J]．中兴通讯技术，2006，12(3)：6-9．

[10] Wang S, Luo F, Ma S. Overview of the Second Generation AVS Video Coding Standard (AVS Coding Standard (AVS2)[J]. ZTECOMMUNICATIONS, 2016,14: 3-11.

[11] Zhang J, Jia C, Lei M, et al. Recent development of AVS video coding standard: AVS3[C]. 2019 picture coding symposium (PCS). IEEE, 2019: 1-5.

[12] Shen L, Liu Z, Zhang X, et al. An effective CU size decision method for HEVC encoders[J]. IEEE transactions on multimedia, 2012, 15(2): 465-470.

[13] Zhao T, Wang Z, Kwong S. Flexible mode selection and complexity allocation in high efficiency video coding[J]. IEEE Journal of Selected Topics in Signal Processing, 2013, 7(6): 1135-1144.

[14] Chiang P T, Chang T S. Fast zero block detection and early CU termination for HEVC video coding[C]. 2013 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, 2013: 1640-1643.

[15] Yong-lin L, Fan L, Li-wei X. A fast CU coding mode decision algorithm for H. 265/HEVC[C]. TENCON 2015 IEEE Region 10 Conference. IEEE, 2015: 1-4.

[16] Wu X, Wang H, Wei Z. Optimal stopping theory based fast coding tree unit decision for high efficiency video coding[C]. 2016 Visual Communications and Image Processing (VCIP). IEEE, 2016: 1-4.

[17] Grellert M, Zatt B, Bampi S, et al. Fast coding unit partition decision for HEVC using support vector machines[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2018, 29(6): 1741-1753.

[18] Zhu L, Zhang Y, Pan Z, et al. Binary and multi-class learning based low complexity optimization for HEVC encoding[J]. IEEE Transactions on Broadcasting, 2017, 63(3): 547-561.

[19] Liu Z, Yu X, Gao Y, et al. CU partition mode decision for HEVC hardwired intra encoder using convolution neural network[J]. IEEE Transactions on Image Processing, 2016, 25(11): 5088-5103.

[20] Zhang Y, Wang G, Tian R, et al. Texture-classification accelerated CNN scheme for fast intra CU partition in HEVC[C]. 2019 Data Compression Conference (DCC). IEEE, 2019: 241-249.

[21] Wang Z, Li F. Convolutional neural network based low complexity HEVC intra encoder[J]. Multimedia Tools and Applications, 2021, 80: 2441-2460.

[22] Li X, Oertel N, Hutter A, et al. Laplace distribution based Lagrangian rate distortion optimization for hybrid video coding[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2008, 19(2): 193-205.

[23] Shajin F H, Rajesh P, Raja M R. An efficient VLSI architecture for fast motion estimation exploiting zero motion prejudgment technique and a new quadrant-based search algorithm in HEVC[J]. Circuits, Systems, and Signal Processing, 2022: 1-24.

[24] Wang H, Wu X, Huang Z, et al. High-frequency component helps explain the generalization of convolutional neural networks[C]. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 8684-8694.

[25] Wu H, Judd P, Zhang X, et al. Integer quantization for deep learning inference: Principles and empirical evaluation[J]. arXiv preprint arXiv:2004.09602, 2020.

[26] Jia C, Wang S, Zhang X, et al. Content-aware convolutional neural network for in-loop filtering in high efficiency video coding[J]. IEEE Transactions on Image Processing, 2019, 28(7): 3343-3356.

[27] 万帅，杨付正．新一代高效视频编码 H.265/HEVC 原理、标准与实现[M]．北京：电子工业出版社，2014：78-82．

[28] 刘畅，贾克斌，刘鹏宇．基于多分支网络的深度图帧内编码单元快速划分算法[J]．电子与信息学报，2022，44(12)：4357-4366．

[29] Filippov V A, Bobylev A N, Busygin A N, et al. A biomorphic neuron model and principles of designing a neural network with memristor synapses for a biomorphic neuroprocessor[J]. Neural Computing and Applications, 2020, 32(7): 2471-2485.

[30] Sara U, Akter M, Uddin M S. Image quality assessment through FSIM, SSIM, MSE and PSNR—a comparative study[J]. Journal of Computer and Communications, 2019, 7(3): 8-18.

[31] Chicco D, Warrens M J, Jurman G. The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation[J]. PeerJ Computer Science, 2021, 7: e623.

[32] Dubey S R, Singh S K, Chaudhuri B B. Activation functions in deep learning: A comprehensive survey and benchmark[J]. Neurocomputing, 2022.

[33] Suvorov R, Logacheva E, Mashikhin A, et al. Resolution-robust large mask inpainting with fourier convolutions[C]. Proceedings of the IEEE/CVF winter conference on applications of computer vision. 2022: 2149-2159.

[34] Tan M, Le Q. Efficientnet: Rethinking model scaling for convolutional neural networks[C]. International conference on machine learning. PMLR, 2019: 6105-6114.

[35] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778..

[36] Liao Y W, Chen M J, Yeh C H, et al. Efficient inter-prediction depth coding algorithm based on depth map segmentation for 3D-HEVC[J]. Multimedia Tools and Applications, 2019, 78: 10181-10205.

[37] Kim K, Ro W W. Fast CU depth decision for HEVC using neural networks[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2018, 29(5): 1462-1473.

[38] Yin P, Zhang L. Image recommendation algorithm based on deep learning[J]. IEEE

Access, 2020, 8: 132799-132807.

[39] Li T, Xu M, Zhu C, et al. A deep learning approach for multi-frame in-loop filter of HEVC[J]. IEEE Transactions on Image Processing, 2019, 28(11): 5663-5678.

[40] Lee W Y, Park S M, Sim K B. Optimal hyperparameter tuning of convolutional neural networks based on the parameter-setting-free harmony search algorithm[J]. Optik, 2018, 172: 359-367.

[41] Santurkar S, Tsipras D, Ilyas A, et al. How does batch normalization help optimization.[J]. Advances in neural information processing systems, 2018, 31.

[42] Ryu J, Yang M H, Lim J. Dft-based transformation invariant pooling layer for visual classification[C]. Proceedings of the European Conference on Computer Vision (ECCV). 2018: 84-99.

[43] Basha S H S, Dubey S R, Pulabaigari V, et al. Impact of fully connected layers on performance of convolutional neural networks for image classification[J]. Neurocomputing, 2020, 378: 112-119.

[44] Shorten C, Khoshgoftaar T M. A survey on image data augmentation for deep learning[J]. Journal of big data, 2019, 6(1): 1-48.

[45] Correa G, Assuncao P, Agostini L, et al. Performance and computational complexity assessment of high-efficiency video encoders[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2012, 22(12): 1899-1909.

[46] Bjontegaard G. Calculation of average PSNR differences between RD-curves[J]. ITU SG16 Doc. VCEG-M33, 2001.

[47] Passos D, Mishra P. A tutorial on automatic hyperparameter tuning of deep spectral modelling for regression and classification tasks[J]. Chemometrics and Intelligent Laboratory Systems, 2022: 104520.

[48] Hassan M, Shanableh T. Predicting split decisions of coding units in HEVC video compression using machine learning techniques[J]. Multimedia Tools and Applications, 2019, 78: 32735-32754.

[49] Agustsson E, Timofte R. Ntire 2017 challenge on single image super-resolution: Dataset and study[C]. Proceedings of the IEEE conference on computer vision and pattern recognition workshops. 2017: 126-135.

[50] Xu M, Li T, Wang Z, et al. Reducing complexity of HEVC: A deep learning approach[J]. IEEE Transactions on Image Processing, 2018, 27(10): 5044-5059.

[51] Tao Y. Image Style Transfer Based on VGG Neural Network Model[C]. 2022 IEEE International Conference on Advances in Electrical Engineering and Computer

Applications (AEECA). IEEE, 2022: 1475-1482.

[52] Li Z, Liu F, Yang W, et al. A survey of convolutional neural networks: analysis, applications, and prospects[J]. IEEE transactions on neural networks and learning systems, 2021:6999-7019.

[53] Feng L, Shu S, Lin Z, et al. Can cross entropy loss be robust to label noise?[C]. Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence. 2021: 2206-2212.

[54] Liu L, Jiang H, He P, et al. On the variance of the adaptive learning rate and beyond[J]. arXiv preprint arXiv:1908.03265, 2019.

[55] Liang J. Confusion Matrix: Machine Learning[J]. POGIL Activity Clearinghouse, 2022, 3(4).

英文缩略词表

英文缩写	英文全称	中文全称
ITU-T	Telecommunication Standardization Sector	国际电信联盟
HEVC	High Efficiency Video Coding	高效视频编解码
CU	Coding Unit	编码单元
ISO	International Organization for Standards	国际标准组织
CTU	Coding Tree Unit	编码树单元
TU	Transform Unit	变换单元
PU	Prediction Unit	预测单元
RDO	Rate Distortion Optimazation	率失真代价
DCT	Discrete Cosine Transform	离散余弦变换
QP	Quantization Parameter	量化参数
CABAC	Context-based Adaptive Binary Arithmetic Coding	自适应二进制算术编码
SAO	Sample Adaptive Offset	自适应补偿
SATD	Sum of Absolute Transformed Difference	残差变换绝对值和
ANN	Artificial Neural Networks	人工神经网络
MSE	Mean Square Error	均方误差
MAE	Mean Absolute Error Loss	平均绝对误差
SVM	Support Vector Machine	向量机
Adam	Adaptive Moment Estimation	自适应时刻估计法
CuprNet	Coding Unit Partition Regression Network	编码单元划分回归网络
CupcNet	Code Unit Partition Classification Network	编码单元划分分类网络
BD-psnr	Bjøntegaard Delta peak signai-to-noise rate	

BD-rate	Bjøntegaard Delta bit-rate	
TS	Time Saver	编码节省时间
SGD	Stochastic Gradient Descent	随机梯度下降法