用于学习视频压缩的时态上下文挖掘

Xihua Sheng、Jiahao Li、Bin Li、Li Li，*IEEE 会员*；Dong Liu，*IEEE 高级会员*；Yan Lu

摘要--近年来，将深度学习*应用*于视频压缩越来越受到关注。在这项工作中，我们解决了端到端学习视频压缩问题，并特别关注如何更好地学习和利用时序上下文。我们建议在获得重建帧之前，不仅要传播最后一帧重建帧，还要传播特征，以进行时间上下文挖掘。我们从传播的特征中学习多尺度时空上下文，并将学习到的时空上下文重新填充到压缩方案的各个模块中，包括上下文编码器-解码器、帧生成器和时空上下文编码器。我们摒弃了对并行化不友好的自回归熵模型，以追求更实用的编码和解码时间。实验结果表明，我们提出的方案实现的压缩率比
现有的学习视频编解码器。我们的方案还优于 x264 和 x265（分别代表 H.264 和 H.265 的工业软件）以及 H.264、H.265 和 H.266 的官方参考软件（分别为 JM、HM 和 VTM）。具体而言，当内部周期为 32 且以 PSNR 为导向时，我们的方案比 H.265-HM 方案节省 14.4% 的比特率；当以 MS-SSIM 为导向时，我们的方案比 H.266-VTM 方案节省 21.1% 的比特率。

索引术语--深度神经网络、端到端压缩、学习视频压缩、时态上下文挖掘、时态上下文重新填充。

I. 导言

如今，视频数据占互联网流量的绝大部分。因此，高效的视频压缩一直是降低传输和存储成本的高要求。在过去的二十年里，已经制定了多个视频编码标准，包括 H.264/AVC [1]、H.265/HEVC [2] 和 H.266/VVC [3]。

最近，学习视频压缩技术又有了新的发展方向。现有的学习视频压缩大致可分为四类。基于残差编码的方案 [4]-[22]：首先在像素域或特征域进行预测。然后对预测的残差进行压缩。基于体积编码的方案 [23]-[25]：

本文于 2022 年 7 月 11 日收到，2022 年 9 月 23 日修订；
2022 年 10 月 29 日接受；当前版本日期为 2023 年 1 月 31 日。

X.Sheng, L. Li, and D. Liu are with the CAS Key Laboratory of Technology in Geo-Spatial Information Processing and Application System, University of China Science and Technology, Hefei 230027, China (e-mail: xhsheng@mail.ustc.edu.cn, lil1@ustc.edu.cn, dongeliu@ustc.edu.cn).

J.Li, B. Li, and Y. Lu are with Microsoft Research Asia, Bei-Jing 100080, China.(e-mail:li.jiahao@microsoft.com, libin@microsoft.com, yanlu@microsoft.com).

本文由副主编 Singh Amit 推荐。(Corre- sponding author: Bin Li.)

这项工作是盛希华在微软亚洲研究院担任全职实习生时完成的。

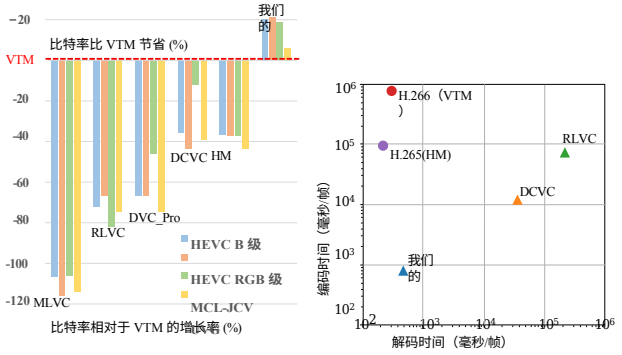


图 1.当内部周期为 32 时，以 MS-SSIM 表示的压缩率
和 1080p 视频的运行时间。由于传统的视频编解码器是针对 CPU 优化的，因此我们报告了它们在 CPU 上的运行时间，这与之前的方案[4]、[5]相同。学习的视频编解码器在 GPU 上运行。

视频被视为包含多个帧的卷。应用三维卷积得到潜在表示。基于熵编码的方案 [26]：每个帧都使用独立的图像编解码器进行编码。探索不同帧中潜在表示之间的相关性，以指导熵建模。基于条件的编码方案 [27]：将时间上下文作为条件，编码器自动探索时间相关性。还有一些其他方案侧重于优化学习编解码器的感知质量 [28]、[29]。我们同意，感知质量更为重要，而且由于端到端训练和反向传播，学习型编解码器在优化感知质量方面比传统编解码器更有优势。不过，本文仍将重点放在客观失真指标上，以便更容易与以前的方案进行比较。

在现有的学习视频编解码器中，Li *等人*提出了深度上下文视频压缩（DCVC）框架，并实现了最先进的压缩率[27]。DCVC 是一种基于上述条件编码结构的开放式框架。DCVC 通过运动补偿和多个卷积层从之前解码的帧中生成单尺度的时间上下文。这是一种简单而实用的方法。但是，由于先前解码的帧只包含三个通道，因此会丢失很多纹理和运动信息。此外，单尺度上下文可能不包含足够的时间信息。因此，在本文中，我们试图找到一种更好的方法来学习和利用时间上下文。

按照条件编码结构，我们提出了一个时态上下文挖掘（TCM）模块，以学习更丰富、更准确的时态上下文。我们在获得重建帧之前传播特征，并学习时态上下文。

利用 TCM 模块从当前帧的传播特征中生成上下文。考虑到学习单尺度上下文可能无法很好地描述时空不均匀性[30]-[32]，TCM 模块采用分层结构生成多尺度时空上下文。最后，我们将学习到的时空上下文重新填充到我们的组合按压方案的各个模块中，包括上下文编码器-解码器、帧生成器和时空上下文编码器。我们将 这一过程称为时空语境再填充（TCR）。

尽管随机存取是最有效的编码配置，但与之前的大多数方案一样， 本文将重点放在有延迟限制或称为低延迟编码的方案上。一些方案报告了比 H.265/HEVC 更高的压缩率，但大多数方案只是使用了 x265 [33]，该方案针对快速编码时间进行了优化，不能代表 H.265/HEVC 的最佳压缩率。在本文中，我们尽力与最好的传统视频编解码器进行比较。我们使用官方参考软件 JM [34]、HM [35] 和 VTM [36]，对 H.264/AVC、H.265/HEVC 和 H.266/VVC 进行编码。以代表最高的潜在压缩比，而不会对其造成任何损害。更多详情可参见第 II-C 章。此外，许多方案 [4], [17], [23], [27], [37] 都使用自回归熵模型 [38] 来提高压缩比，这给并行实施带来了巨大挑战，同时也大大增加了解码时间。尽管我们认为自回归熵模型可以进一步大幅提高压缩比，但我们并没有在我们方案的任何部分应用自回归熵模型来构建一个便于并行化的解码器。图 1 比较了 1080p 视频的压缩率和运行时间。与其他已学习的视频编解码器相比，我们的方案实现了更快的编码和解码速度。据我们所知，这是首个端到端学习视频压缩方案取得如此里程碑式的成果，在峰值信噪比（PSNR）方面优于 HM，在多尺度结构相似性指数测量（MS-SSIM）方面优于 VTM [39]。

我们的贡献概述如下

- 我们提出了一个时间上下文挖掘模块，从传播的图像而不是之前重建的帧中学习多尺度时间上下文。
- 我们建议将学习到的多尺度时间上下文重新填充到上下文编码器-解码器、帧生成器和时间上下文编码器中，以帮助压缩和重建当前帧。
- 在没有自动回归熵模型的情况下，我们提出的方案比现有的学习视频编解码器实现了更高的压缩率。在 PSNR 方面，我们的方案也比 H.265/HEVC-HM 的参考软件高出 14.4%；在 MS-SSIM 方面，我们的方案比 H.266/VVC-VTM 的参考软件高出 21.1%。我们将发布代码，以方便未来的研究。

本文的其余部分安排如下。第二节简要回顾了相关工作。第三节介绍了我们提出的方案的方法。第四节介绍了实验结果和对所提方案的分析。第五节总结本文并讨论未来研究。

II. 相关工作

A. 传统视频压缩

传统的视频压缩技术已经发展了几十年，并提出了多个视频编码标准。H.264/AVC [2] 最初由著名的 ITU- T 和 ISO/IEC 标准在 1999 年至 2003 年期间制定。它取得了巨大的成功，被广泛应用于高清电视信号广播、互联网和移动网络视频等许多领域。随着视频分辨率的提高和并行处理架构的使用，H.265/HEVC [1] 于 2013 年定型，比 H.264/AVC 节省了约 50%的比特率。H.265/HEVC 优越的压缩效率使 4K 视频的保真度得以提高，并得到普及。H.266/VVC [3] 是新一代国际视频编码标准。与 H.265/HEVC 相比，它不仅能大幅降低比特率，还能满足当前和新兴媒体的所有需求。这些视频编码标准遵循类似的混合视频编码框架，包括预测、变换、量化、熵编码和循环过滤。尽管学习型图像编码解码器 [40], [41] 已经赶上甚至超过了传统的图像编码方案，但就压缩比而言，传统的视频压缩方案仍然是最先进的。

B. 学习视频压缩

近年来，学习视频压缩探索出了一个新方向。Lu 等人用卷积神经网络取代了传统的运动补偿预测和残差编码框架中的每个部分 [4]。他们利用速率-失真成本对整个网络进行联合运算。Lin 等人探索了多个参考帧和相关的多个运动矢量，以生成更精确的当前帧预测并降低运动矢量的编码成本 [9]。Yang 等人设计了一种递归学习视频压缩方案 [16]。提出的递归自动编码器和递归概率模型利用大范围帧的时间信息生成潜在表示并重建压缩输出。Hu 等人提取了输入帧的特征，并应用可变形卷积进行运动预测[10]。学习到的偏移和残差在特征域中被压缩。然后通过非局部注意模块融合解码缓冲区中存储的多个参考特征，以获得重构帧。除了运动补偿预测和残差编码框架外，Habibian 等人将视频视为由多个帧组成的体，并提出了一种直接压缩多个帧的三维自动编码器[23]。Liu 等人使用图像编解码器压缩每个帧，并提出了一种熵模型来探索潜在表示的时间相关性 [26]。Li 等人将范式从残差编码转向了条件编码 [27]。他们从之前解码的帧中学习时间上下文，让编码器探索时间相关性，从而自动去除冗余。

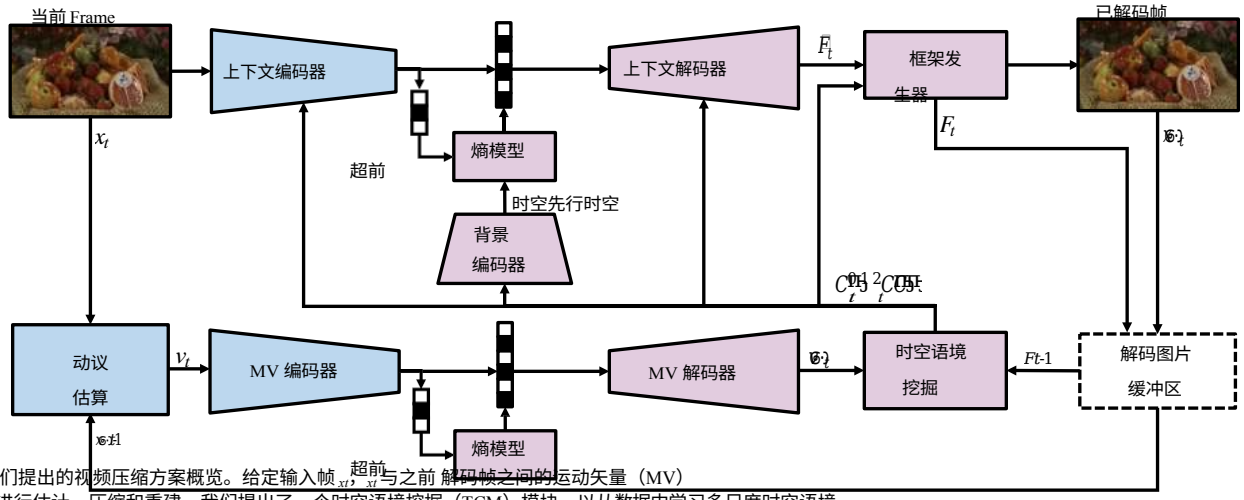


图 2.我们提出的视频压缩方案概览。给定输入帧 x_t，超前 x_t 与之前 解码帧之间的运动矢量（MV）$\hat{x}_{t-1}$ 进行估计、压缩和重建。我们提出了一个时空语境挖掘（TCM）模块，以从数据中学习多尺度时空语境。传播的特征 F_{t-1}，而不是先前解码的帧 $\hat{x}_{t-1}$。时间上下文 (C_t^0, C_t^1, C_t^2) 被重新填入上下文编码器-解码器、帧生成器和时序上下文编码器用于压缩和重建当前帧 x_t。 解码后的帧 $\hat{x}_t$ 和获得 $\hat{x}_t$ 之前的特征 F_t 将被传播，以帮助压缩下一帧 x_{t+1}。蓝色模块仅用于编码器端。淡紫色模块同时用于编码器和解码器两侧。

C. 传统视频编解码器与学习型视频编解码器的比较

虽然许多现有的学习视频编解码器都 报告了比 H.265/HEVC 更好的压缩率，但它们可能并不

使用最佳编码器存在以下主要问题。首先，x264 [42] 和 x265 [33] 分别用来表示 H.264/AVC 和 H.265/HEVC 的压缩率。然而，它们是为了更快的编码而不是更高的压缩比而优化的。应使用官方参考软件 JM [34]、HM [35] 和 VTM [36] 来表示更高的压缩比。例如，x265 与 HM 之间的差距约为 80%，如表 I 所示。其次，为了限制时间误差传播，使用了 10 或 12 个不切实际的内部周期[43]。然而，在实际应用中，当使用如此小的内部周期时，I 帧的比特占总比特数的 很大一部分。以 VTM 为例，I 帧的比特在周期 12 中占 51%，在周期 32 中占 29%。周期内 12 对压缩比有害，在实际应用中很少使用。因此，本文建议使用周期内 32（30fps 约为 1 秒）来衡量视频编解码器的间隔编码性能更为合理。第三，现有方案为传统编解码器设置了 YUV420 内部色彩空间。然而，在比较了传统编解码器的不同色彩空间后，我们得出结论：尽管最终失真度是以 RGB 测量的，但 YUV444 优于 YUV420 和 RGB。性能差距超过 30%。因此，我们建议使用 YUV444 作为 JM、HM 和 VTM 的内部色彩空间，而不是寻求传统编解码器更高的潜在压缩比。在学习图像压缩方面也有类似的结论[40]。第四，我们为传统编解码器使用最合适的配置（例如，使用范围扩展配置文件对 YUV444 内容进行编码），因为不同配置的性能差距可能超过 40%。

III. 方法

A. 概述

旨在找到学习和利用时间的更好方法

因此，我们提出了一种基于时空上下文挖掘和重新填充的新型学习视频压缩方案。图 2 是我们的方案概览。

*1) 运动估计：*我们将当前帧 x_t 和之前解码的帧 $\hat{x}_{t-1}$ 输入基于神经网络的运动估计模块，以估计光流。光流被视为每个像素的估计运动矢量 (MV) v_t。本文的运动估计模块基于预训练的 Spynet [44]。

*2) MV 编码器-解码器：*获得运动向量 v_t 后，我们使用 MV 编码器和解码器以有损方式压缩和重建输入 MV v_t。具体来说，v_t 是由超先验 结构的自动编码器压缩的[45]。重构后的 MV 表示为 $\hat{v}_t$。

*3) 学习时间上下文：*我们提出了一个 TCM 模块，从传播的特征 F_{t-1} 而不是之前解码的帧 $\hat{x}_{t-1}$ 中学习更丰富、更准确的时间上下文。TCM 模块不再只生成单尺度的时间上下文，而是生成多尺度的时间上下文。

文本 C_t^{-l} 来捕捉空间-时间非均匀运动和纹理，其中 l 是不同尺度的指数。学习到的时空上下文被重新填充到上下文编码器-解码器、帧生成器和时空上下文编码器中，以帮助提高压缩比。这一过程被称为 TCR。我们将在 第 III-B 和 III-C 节中详细介绍它们。

*4) 上下文编码解码器和帧生成器：*在重新填充的多尺度时空文本 C_t^{-l} 的帮助下，上下文编码器-解码器和帧生成器将对时间序列进行编码和解码。

用于压缩和重建当前帧。

x_t。我们将解码后的帧表示为 $\hat{x}_t$，并将之前的特征表示为

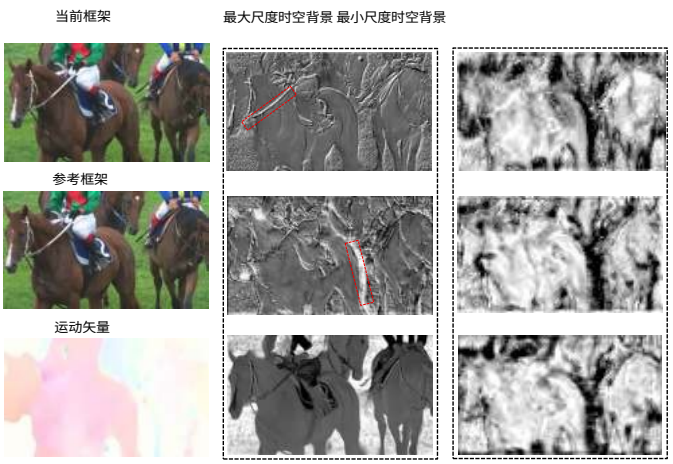


图 3. 多尺度时空背景的可视化。在最大尺度上 下 文 中，一些通道关注纹理信息，一些通道关注颜色信息。在最小尺度上下文中，通道主要关注运动量大的区域。

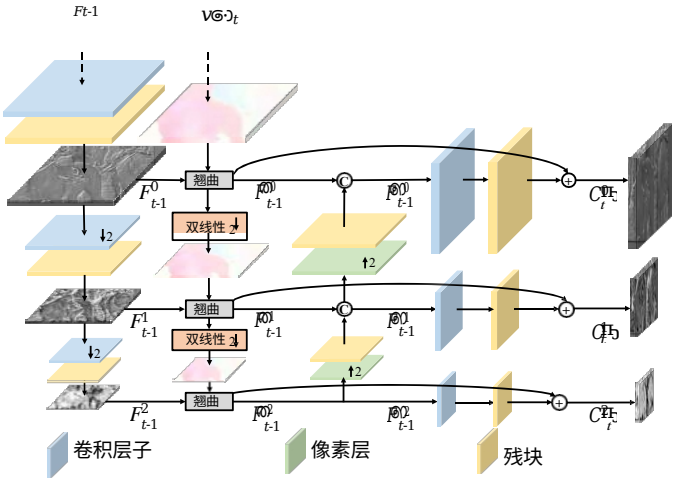


图 4. 时空语境挖掘（TCM）模块的架构。传播的特征 F_{t-1} 输入 TCM 模块，生成多尺度的
时间背景 C_t .

$_t$x̂ $_t$ 和 F_t 被传播，以帮助压缩下一帧 x_{t+1} 。详情见第 III-C 节。

5) 时态上下文编码器： 为了利用上下文编码器生成的不同帧的潜在表示的时间相关性，我们使用时间上下文编码器来生成时间先验，利用
多尺度时空背景 C^{-l} 。更多信息见第 III-C$_t$ 节。

6) 熵模型： 我们使用因子化熵模型
超先验分布和拉普拉斯分布，以模拟潜
[27]。尽管自动回归熵模型有助于提高压缩率，但我们并没有使用该模型来使解码过程更易于并行化。
比。对于上下文编码器-解码器，我们融合了超先验和时间先验[27]，以估计出更精确的参数
的拉普拉斯分布。在写入比特流时，我们使用
与 [46] 中的实现方法类似。

B. 时空语境挖掘

考虑到先前解码的帧 x̂ $_{t-1}$ 只包含三个信道，因而损失了很多信息，因此并不
从 x̂ $_{t-1}$ 中学习时间上下文的最佳方法。在本文中，我们提出了一个 TCM 模块，从传播的特征 F_{t-1} 中学习时间上下文。不同于现有的特征域视频压缩方案 [10]、[11]、[47]，这些方案都是以时间为基础的。
从先前解码的帧 x̂ $_{t-1}$ 中提取特征，我们在获得 x̂ $_{t-1}$ 之前传播特征 F_{t-1} 。具体来说
在 x̂ $_{t-1}$ 的重构过程中，我们将帧生成器最后一个卷积层之前的特征 F_{t-1} 保存在广义解码图片缓冲区（DPB）中。为了降低计算复杂度，我们不存储多个特征，而是将其存储在一个通用的解码图片缓冲区（DPB）中。
的解码帧[10]、[11]，我们只存储单个

此外，学习单一尺度的上下文可能无法很好地描述时空不均匀的运动和纹理[30]-[32]。如图 3 所示，在最大尺度上下文中，一些通道关注纹理信息，一些通道关注颜色信息。而在最小尺度的情况下，通道主要关注运动量大的区域。因此，我们采用分层方法来学习多尺度时间上下文。

如图 4 所示，我们首先生成多尺度特征
$F^{l\text{t-1}}$ 从传播的特征 F_{t-1} 使用特征前
L 级牵引模块（*提取*），包括
卷积 层和残差块 [48]（本文中使用了三个层次）。

$$F^{l\text{t-1}} = extract\,(F_{t-1})\,,\ l = 0,\ 1,\ 2 \tag{1}$$

同时，对解码后的 MV v̂ $_t$ 进行降采样。
双线性滤波器生成多尺度 MV v̂ ^{l} ，其中 v̂^{0} 是
设为 v̂ $_t$ 。请注意，每个下采样 MV 都要除以 2。
然后，我们对多尺度特征 F^l t-1 使用
相关的 MV v̂^{l} 。 $_t$

$$\bar{F}^{l\text{t-1}} = warp\,F^{l}_{\text{t-1}}\,,\ \hat{\mathrm{v}}^{l}_{t}\,,\ l = 0,\ 1,\ 2 \tag{2}$$

之后，我们使用由一个子像素层 [49] 和一个残差块组成的上采样（*upsample*）模块对 $\bar{F}^{l+1}$ 进行上采样。然后将上采样后的特征串联起来 $_{\text{t-1}}$(*concat*)
与 $\bar{F}^{l}$ $_{\text{t-1}}$ 在同一尺度上。

$$\tilde{F}^{l\text{t-1}} = concat\left(\bar{F}^{l}_{\text{t-1}},\ upsample\left(\bar{F}^{l+1}_{\text{t-1}}\right)\right),\ l = 0,\ 1 \tag{3}$$

在分层结构的每一层，由一个卷积层和一个残差块组成的语境细化模块（*细化*）用于学习残差[50]。残差被添加到 $\bar{F}^{l}$ ，以 生成最终的时空语境

$^{-}{}_{l}$ t-1

特征。然后传播 F_{t-1} ，学习压缩当前帧 $\hat{x}_t$ 的时间上下文。对于第一个 P 帧、

我们仍然从重建的 I 帧中提取特征，以使模型适应不同的图像编解码器。

C_t ，如图 4 所示。

$$\bar{C}_t^l = \bar{F}_{t-1}^l + refine\left(F_{t-1}^l\right), l = 0, 1, 2 \quad (4)$$

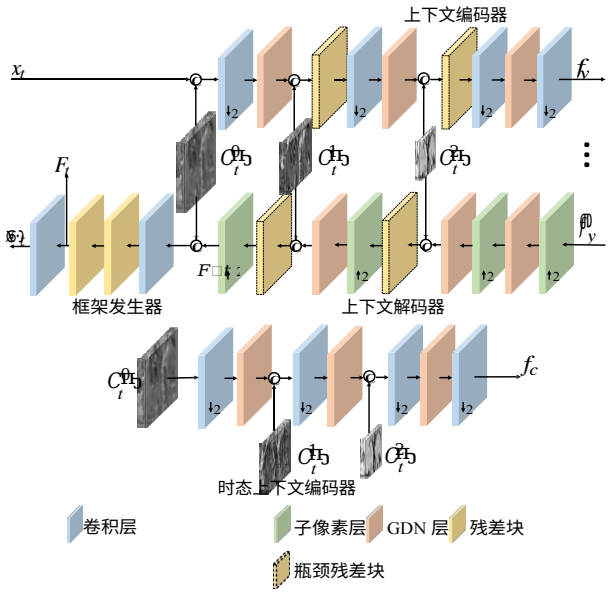


图 5.时空语境重新填充（TCR）示意图。 多尺度时空上下文被重新填充到上下文编码器-解码器、帧生成器和时空上下文编码器中。

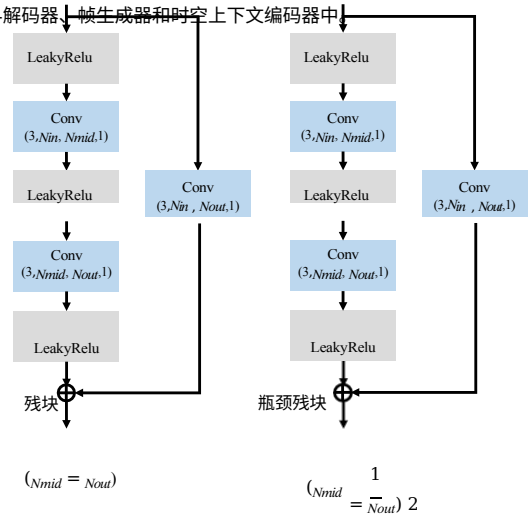


图 6.残差块和瓶颈残差块的结构。卷积层中的数字（3, N_{in} , N_{mid} ， 1）表示内核大小为 3，输入通道数为 N_{in} ，输出通道数为 N_{mid} ，跨距为 1。

C. *时间背景重新填充*

为了充分利用时空相关性，我们将学习到的多尺度时空上下文重新填充到压缩方案的各个模块中，包括上下文编解码器、帧生成器和时空上下文编码器，如图 5 所示。时空上下文在时空预测和时空熵建模中发挥着重要作用。有了重新填充的多尺度时空上下文，我们方案的 压缩率得到了很大提高。

1) 上下文编码解码器和帧生成器：

我们将最大尺度的时间上下文 C_t^{-0} 与 x_t 然后将其输入上下文编码器。在从 x_t 映射到潜在的表示 f_y ，我们还将 C_t^{-1} 和 C_t^{-2} 与其他将刻度输入编码器。与编码器对称的是上下文解码器映射量化的潜在表示 $\hat{f}_y$ 在 C_t^{-1} 和 C_t^{-2} 的帮助下，将 $\hat{f}_y$ 转换为特征 $\hat{F}_t$ 。那么 $\hat{F}_t$ 和 C_t^{-0} 汇总后输入帧发生器得到重建帧 $\hat{x}_t$ 。考虑到由于连接会 增加信道数量，因此使用 "瓶颈 "建筑残块来降低中间层的复杂性。具体架构如图 6 所示。在帧生成器的最后一个卷积层之前的特征 F_t 被传播，以帮助压缩下一个 帧 x_{t+1} 。

2) 时间上下文编码器： 为了探索不同帧的潜在表征的时空相关性，我们使用时空上下文编码器来获取低维时空先验 f_c 。而不是使用单一的时空上下文编码器。

与 [27]一样，我们将多尺度的时空背景串联起来。
在生成时空先验的过程中，需要考虑时空背景。时空先验与超先验融合，以估计潜表征 $\hat{f}_y$ 的拉普拉斯分布的均值和方差。时态上下文编码器的结构如图 5 所示。

D.*损失函数*

我们的方案旨在共同优化速率-失真（R-D）成本。

$$L_t = \lambda D_t + R_t = \lambda d(x_t , \hat{x}_t) + R_{\hat{v}_t} + R_{\hat{f}_t} \quad (5)$$

L_t 是当前时间步长 t 的损失函数。$d(x_t , \hat{x}_t)$ 指输入帧 x_t 与重建帧 $\hat{x}_t$ 之间的失真，其中 d(-) 表示均方误差或 1-MS-SSIM[39]。$R_{\hat{v}_t}$ 表示用于编码量化运动矢量潜在表示的比特率，R(-)表示平均平方误差或 1-MS-SSIM[39]。$R_{\hat{f}_t}$ 表示用于相关的超先验。对量化的上下文潜在表示和相关超先验进行编码。与之前的方案[9]一样，我们逐步训练模型，使训练更加稳定。

值得一提的是，在过去五个纪元中，我们使用了在最近的论文[8]、[16]中，这是一种常用的训练策略、[43]，[51]，在连续的训练帧上训练我们的模型，以减少误差传播。

$$L^T = \frac{1}{T}\sum_t L_t = \frac{1}{T}\sum_t \lambda d(x_t , \hat{x}_t) + R_{\hat{v}_t} + R_{\hat{f}_t} \quad (6)$$

其中，T 是时间间隔，在我们的实验中设为 4。

IV. 实验

A. *实验装置*

1) 数据集： 许多训练数据集都是为学习视频压缩而制作的[53]-[55]。在我们的工作中，我们沿用了之前的大多数方案 [4]、[9]、[16]、[27]，使用了常用的 Vimeo-90k [55] 训练数据集，并将视频随机裁剪为 256 × 256 补丁。为了评估我们提出的视频识别技术的性能和应用领域，我们采用了以下方法压缩方案，我们使用了 UVG [56]、MCL-JCV [57] 和 HEVC 数据集 [2]。这些数据集包含各种视频内容，包括慢动作/快动作和劣质/高质量视频，这些内容

videoSRC18_1920x1080_25 videoSRC20_1920x1080_25

videoSRC24_1920x1080_24 videoSRC25_1920x1080_24

图 7.被 MCL-JCV-26 数据集排除在外的四个 动画序列（18、20、24、25）的可视化。

鸭子和腿 EBULupoCandlelight EBURainFruits

和服 OldT ownCross 公园场景

图 8.HEVC Class RGB 包含 6 个 1080p 视频序列，深度为 10 比特，格式为 RGB444，定义在 HEVC 范围扩展的通用测试条件中[52]，包括 *DucksAndLegs*、*EBULupoCandlelight*、*DucksAndLegs*、*DucksAndLegs*、*Kimono*、*OldTownCross* 和 *ParkScene*。

是精益视频压缩中常用的压缩技术。UVG 数据集有 7 个 1080p 序列，MCL-JCV 数据集有 30 个 1080p 序列。考虑到 MCL-JCV 中的动画序列（18、20、24、25）与训练数据集中的自然视频有很大不同，如图 7 所示，我们按照文献[12]中的设置，建立了另一个 MCL-JCV-26 数据集，其中不包括这四个动画序列。这些数据集的源视频都是 YUV420 格式，而 PSNR 是在 RGB 色彩空间中计算的。因此，我们建立了一个名为 HEVC Class RGB 的额外类别，其中包括 HEVC 范围扩展通用测试条件[52]中定义的 6 个 1080p 10 比特深度的 RGB444 格式视频：如图 8 所示：*DucksAndLegs*、*EBULupoCandlelight*、*DucksAndLegs*、*DucksAndLegs*、*Kimono*、*OldTownCross* 和 *ParkScene*。

*2) 实施细节：*我们使用不同的 λ 值（$\lambda = 256、512、1024、2048$）训练四个模型，以实现多种编码率。在使用 MS-SSIM 进行性能评估时，我们使用 1-MS-SSIM 作为失真损失，以不同的 λ 值（$\lambda = 8、16、32、64$）对模型进行微调。我们的模型使用 PyTorch 实现，并在 2 个英伟达 V100 GPU 上进行训练。训练模型需要 2.5 天。

*3) 评估指标：*我们使用 bpp（每像素比特）来衡量每个帧中一个像素的比特成本。我们使用 PSNR 和 MS-SSIM 来评估解码帧与原始帧之间的失真。

B. 实验结果

*1) 比较设置：*尽管随机存取是最有效的编码模式，但本文仍沿用了之前大多数方案的设置，将重点放在低延迟编码模式上。因此，我们将 x264、x265、JM-19.0、HM-16.20 和 VTM-13.2 设置为低延迟模式。对于 x264 和 x265，我们使用 FFmpeg 在 *veryslow* 预设下的实现。与之前的方案一样，内部色彩空间为 YUV420。对于 JM、HM 和 VTM，我们使用范围扩展配置文件而非主配置文件，以启用为非 YUV420 设计的编码工具。具体来说，我们分别使用*编码器 JM LB___ HE*，*编码器_ lowdelay_ main_ rext*，*编码器_ lowdelay_* vtm configu-ration。内部色彩空间设置为 YUV444。同时，为寻求最高压缩比，默认使用四个参考帧。正如第 II-C 节所分析的，I 帧的比特占总比特数的很大一部分。因此，对于已学习的视频编解码器（DVC_ Pro [4]、MLVC [9]、RLVC [16]、DCVC [27]），我们使用的是

为了进行公平比较，我们采用了与我们的方案相同的图像编解码器来压缩 I 帧。我们采用由 CompressAI [46] 实现的 *Cheng2020Anchor* [41]，但去掉了它的自动回归熵模型 [38]。DVC_ Pro 模型是我们自己训练的，其他模型是作者发布的。在以前的方案中，内部周期被设置为 10 或 12，以限制时间误差的传播。然而，正如第二章 C 节所述，在实际应用中很少使用这么小的内部周期。因此，我们建议使用更合理的 32 内周期。为了提供更多信息，我们还测试了内部周期为 12 的情况。由于带有*编码器_ lowdelay_* vtm 配置的 VTM 的 GOP 大小为 8，因此不支持周期内 12。因此，我们添加了一个改进的 VTM*，它具有低延迟 P、8 位内部深度和单参考帧配置。在所有测试数据集中，我们为每个 视频编码 96 个帧。

*2) 结果*表 I 和表 II 报告了内部周期设置为 32 时的 BD 速率[59]。表 III 和表 IV 报告了内部周期设为 12 时的 BD 速率。锚点为 HM。与 HM 相比，负值表示比特率节省，正值表示比特率增加。实验结果表明，对于周期内 32，我们的方案在 HEVC Class RGB 的 PSNR 方面优于 HM 14.4%，优于 VTM* 14.2%。虽然在最佳配置下，我们的方案与 VTM 相比仍有 25.4% 的 PSNR 性能差距，但在 MS-SSIM 方面，我们的方案比 VTM 优胜 21.1%。在 12 期内，我们实现了更大的压缩率提升。需要注意的是，在将我们的方案与 VTM 或 VTM* 进行比较时，VTM 或 VTM* 被视为锚点，BD 速率值需要重新计算，而不是直接减去上述表格中的值。与第 32 期和第 12 期的结果相比，我们的方案仍然获得了较高的比较率，而其他已学习视频编解码器的比较率则大大降低。

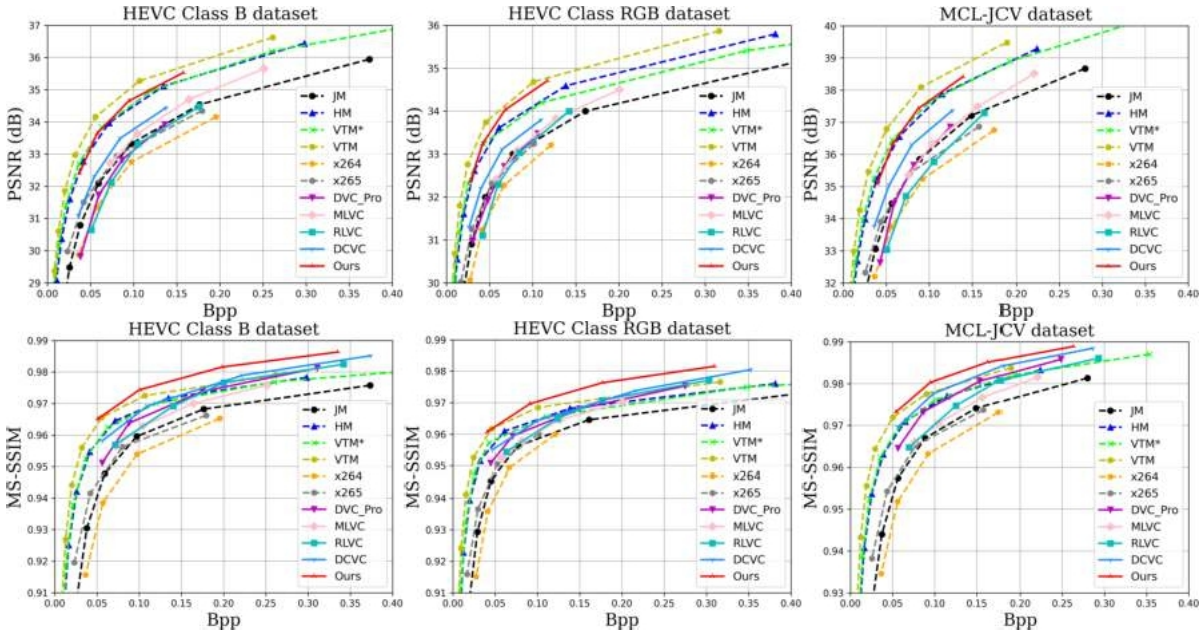


图 9.我们提出的方案在 HEVC Class B、HEVC Class RGB 和 MCL-JCV 数据集上的速率失真性能。 设置为 32。

表 I
PSNR 的 BD 速率（周期内 32）。锚点为 HM。

	HM	JM	VTM	VTM*	x264	x265	DVC Pro	MLVC	RLVC	DCVC	我们的
UVG	0.0	108.1	-28.9	4.6	176.8	109.2	137.7	66.5	140.1	67.3	-9.0
MCL-JCV	0.0	95.4	-31.2	-7.2	143.3	84.4	99.3	66.8	124.8	42.8	-3.2
MCL-JCV-26	0.0	101.9	-31.0	-6.4	161.0	93.6	92.0	56.1	115.8	37.3	-9.7
HEVC B级	0.0	96.9	-28.8	-8.2	144.6	76.1	123.7	61.4	122.6	56.0	-5.3
HEVC C级	0.0	56.6	-29.0	-3.8	79.8	46.2	124.0	124.1	118.9	76.9	15.1
HEVC D级	0.0	50.0	-26.5	-3.5	72.0	43.8	93.6	96.1	81.2	52.8	-5.4
HEVC E级	0.0	80.5	-29.1	-10.0	153.2	60.3	283.0	138.8	246.2	156.8	18.5
HEVC RGB级	0.0	102.4	-29.7	-1.7	151.9	82.8	102.1	82.1	114.2	51.9	-14.4

除非另有说明，我们将 JM、HM 和 VTM 配置为低延迟编码的最高压缩比设置。
‡VTM* 使用一个参考帧，而不是 VTM 默认的四个参考帧。请注意，我们的方案使用一个参考帧进行运动估计。

表 II
MS-SSIM 的 BD 速率（周期内 32）。锚点为 HM。

	HM	JM	VTM	VTM*	x264	x265	DVC Pro	MLVC	RLVC	DCVC	我们的
UVG	0.0	105.6	-27.0	2.3	169.9	87.9	36.2	64.7	49.4	9.2	-25.5
MCL-JCV	0.0	108.5	-30.4	-5.8	141.0	71.9	7.8	50.3	34.5	-16.3	-38.3
MCL-JCV-26	0.0	113.4	-30.3	-6.5	152.0	74.3	0.7	44.2	23.3	-18.8	-40.5
HEVC B级	0.0	112.4	-26.9	-4.2	150.3	71.1	23.5	50.2	28.3	0.9	-40.8
HEVC C级	0.0	61.3	-27.9	-3.4	89.9	53.5	17.0	53.1	30.0	-8.9	-42.4
HEVC D级	0.0	52.4	-25.9	-3.0	80.0	49.7	-7.8	40.4	0.2	-24.2	-52.6
HEVC E级	0.0	90.9	-27.8	-7.8	184.5	55.0	110.1	106.1	87.1	38.0	-40.9
HEVC RGB级	0.0	107.4	-27.2	-3.9	135.9	64.9	18.3	51.8	21.5	3.3	-43.4

内周期较大。图 9 和图 10 展示了 HEVC Class B、HEVC Class RGB 和 MCL-JCV 不同内部周期的 RD- 曲线。我们发现，当源视频为 RGB 格式时，学习的视频编解码器表现更好。图 11 所示的主观比较结果表明，我们的方案可以保留更多细节。

*3) 模型复杂度和编解码时间：*我们的方案参数总数为 10.7M。处理一个 1080p 帧时，我们的方案的 MAC（乘积运算）为 2.9T，而 DCVC 为 2.4T.图 1 显示了不同编解码器对一个 1080p 帧的平均编码和解码时间。表 V 列出了 详细数值。与不包括熵编码时间[16]、[26]、[27]或 CPU 和 GPU 之间数据传输时间[5]的现有方案不同，我们包括了模型推理、熵建模、熵解码以及 CPU 和 GPU 之间数据传输的时间。我们的编码时间指的是从读取原始帧到将比特流写入硬盘的所有时间，而我们的解码时间指的是从读取比特流到生成比特流的所有时间。

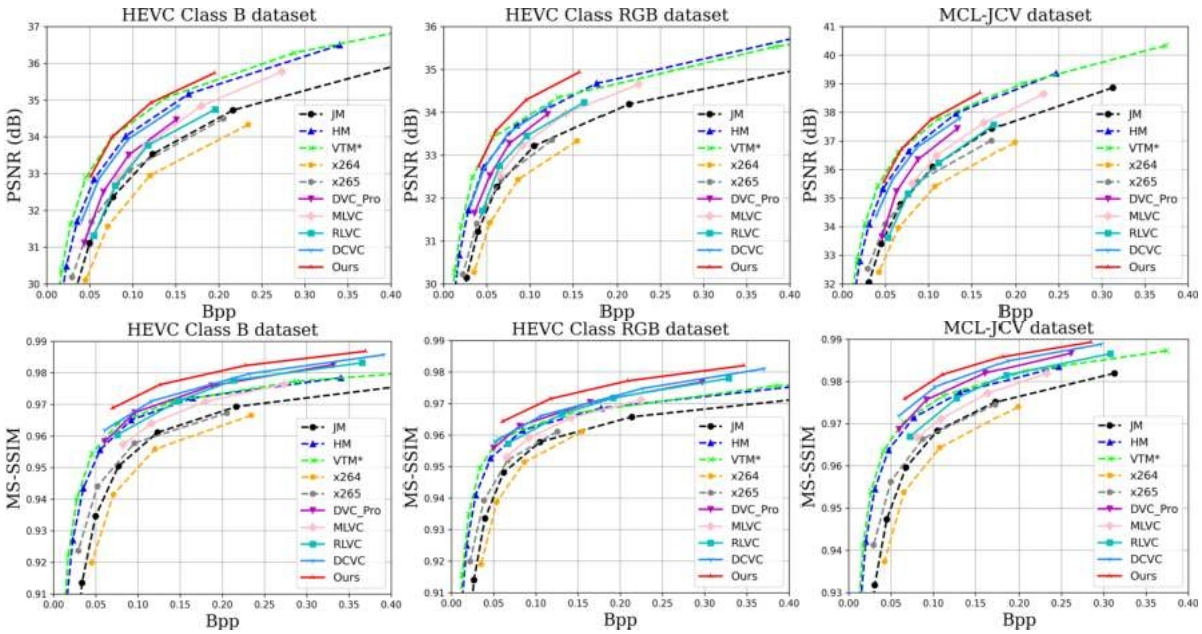


图 10.我们提出的方案在 HEVC Class B、HEVC Class RGB 和 MCL-JCV 数据集上的速率失真性能。 中周期设置为 12。

表 III
PSNR 的 BD 速率（周期内 12）。锚点为 HM。

	HM	JM	VTM*	x264	x265	DVC Pro	MLVC	RLVC	DCVC	我们的
UVG	0.0	79.8	-7.6	139.2	77.7	43.5	37.8	63.2	13.8	-21.0
MCL-JCV	0.0	80.0	-13.3	124.4	67.8	38.7	46.4	73.6	10.0	-12.5
MCL-JCV-26	0.0	84.5	-12.9	138.8	74.8	32.9	37.1	65.1	4.4	-18.8
HEVC B级	0.0	75.8	-16.8	113.3	53.7	36.9	34.6	54.0	9.5	-15.2
HEVC C级	0.0	46.0	-11.7	62.7	31.3	46.3	78.1	58.9	26.1	4.7
HEVC D级	0.0	40.9	-10.4	56.3	29.3	30.9	61.1	35.4	12.7	-10.2
HEVC E级	0.0	62.7	-18.3	113.8	39.3	79.2	54.7	60.4	33.8	-6.4
HEVC RGB 级	0.0	78.7	-11.7	118.8	60.2	24.2	42.4	38.7	2.2	-23.0

这里没有测试为低延迟编码设置最高压缩比的 †VTM，因为它不支持周期内 12。

表 IV
MS-SSIM 的 BD 速率（12 期内）。锚点为 HM。

	HM	JM	VTM*	x264	x265	DVC Pro	MLVC	RLVC	DCVC	我们的
UVG	0.0	79.9	-9.3	132.5	59.5	-2.9	39.4	16.0	-16.1	-35.0
MCL-JCV	0.0	91.4	-12.3	121.8	54.2	-17.0	33.6	10.5	-30.7	-44.4
MCL-JCV-26	0.0	94.7	-13.1	129.8	54.9	-21.3	28.5	1.6	-32.8	-46.5
HEVC B级	0.0	90.2	-13.0	118.6	49.4	-14.7	27.0	-3.7	-24.2	-48.5
HEVC C级	0.0	49.7	-11.9	70.6	36.7	-17.5	27.3	-4.4	-29.8	-47.2
HEVC D级	0.0	41.8	-9.6	62.2	34.9	-31.4	21.2	-24.7	-39.7	-55.1
HEVC E级	0.0	69.7	-14.5	138.1	34.6	5.0	38.8	-5.3	-19.5	-53.9
HEVC RGB 级	0.0	84.6	-12.8	103.4	43.5	-20.3	20.8	-11.2	-25.1	-51.4

最后的重建帧。由于传统视频编解码器针对 CPU 进行了优化，因此我们在英特尔（R）Xeon（R）Platinum 8272CL CPU 上运行这些编解码器，这与之前的方案[4]、[5]相同。对于学习的视频编解码器，我们在英伟达 V100 GPU 上运行。我们只与已学习的视频编解码器进行比较，这些编解码器的官方发布代码支持比特流写入。比较结果表明，我们的方案编码 1080p 帧平均耗时 0.88 秒，解码 1080p 帧平均耗时 0.47 秒。虽然与 DCVC 相比，我们的方案的 MACs 增加了约 21%，但由于我们去除了并行化------，解码时间减少了约 98.7%。

尽管自动回归熵模型有助于提高压缩性能，但它对我们来说并不友好。与 HM 和 VTM 相比，虽然我们的方案实现了更少的编码时间和相似的编码时间，但这是因为它是在非常复杂和昂贵的 GPU 上运行的，而 HM 和 VTM 是在通用 CPU 上运行的。因此，我们方案的复杂性，尤其是解码复杂性，还需要进一步降低，才能投入实际应用。

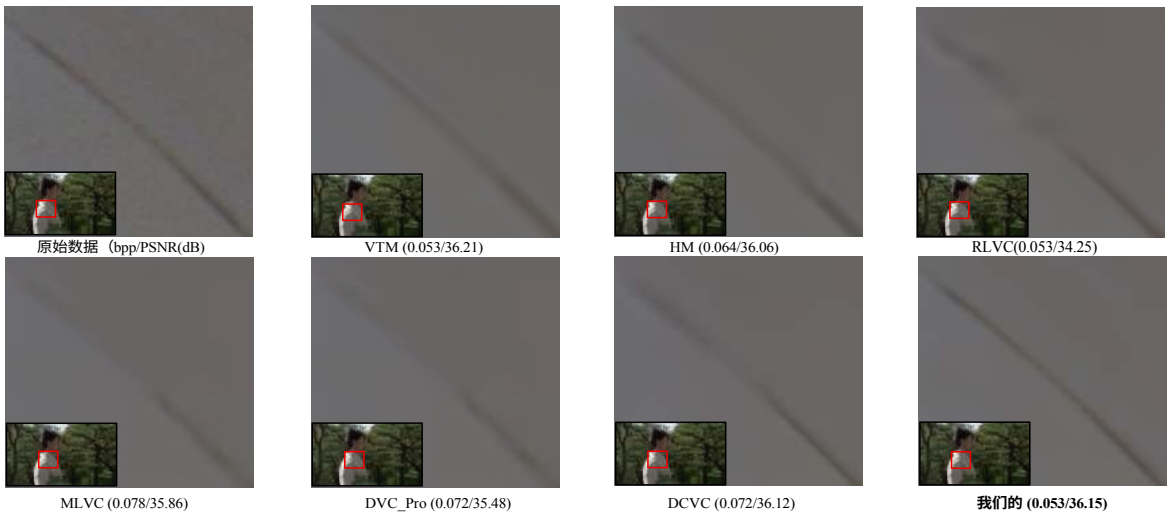


图 11.当内部周期设置为 32 时，HEVC B 级*和服*序列第 3 帧的主观质量对比。

表 V
1080P 帧的平均编码/解码时间（秒）。

计划	Enc 时间	12 月 时间
VTM	743.88 s	0.31 s
HM	92.58 s	0.21 s
DCVC	12.26 s	35.59 s
RLVC	72.00 s	216.67 s
我们的	0.88 s	0.47 s

表 VI
我们计划不同组成部分的有效性。

FP	中药	TCR	B	C	D	E	RGB
✓	✓	✓	0.0	0.0	0.0	0.0	0.0
✓	✓	-	3.5	4.2	3.7	1.8	1.8
-	✓	✓	4.9	6.2	5.6	6.5	3.0
-	✓	-	7.5	7.7	7.9	7.5	4.7
✓	-	-	9.4	11.0	12.1	5.0	3.6
-	-	-	11.6	13.7	15.2	9.5	5.8

表 VII
时间背景对不同共同 MPONENTS 的影响。

	B	C	D	E	RGB
无编码器背景	28.8	36.3	28.6	14.7	9.3
解码器中无上下文	3.5	3.2	3.4	2.4	1.5
不带背景的框架式发电机	8.1	9.8	13.0	8.0	6.6
熵模型中不含上下文	5.7	5.3	5.5	7.8	4.7

表 VIII
中药的层数和重新填充的时间语境的影响。

	B	C	D	E	RGB
3L3C	0.0	0.0	0.0	0.0	0.0
4L1C	3.6	4.4	3.6	1.5	1.5
3L1C	3.5	4.2	3.7	1.8	1.8
2L1C	5.4	6.4	5.8	3.6	2.7
4L4C	0.7	1.2	1.0	0.8	-0.2
3L2C	1.8	1.9	1.1	1.0	0.4
2L2C	3.7	4.0	3.6	2.5	2.2

C. *消融研究*

*1) 建议的不同组件的有效性：*我们的工作重点是更好地学习和利用时间上下文。我们建议从传播的特征中学习时间上下文，并将学习到的时间上下文重新填充到压缩方案的模块中。为了验证这些想法的有效性，我们进行了表 VI 所示的消减研究，其中基线为我们的最终解决方案（即特征传播 (FP) + 时态上下文挖掘 (TCM) + 时空上下文重新填充 (TCR)）。从表中我们可以发现，从重建帧或传播特征中学习和重新填充时空上下文可以提高压缩率。这验证了所提出的 TCM 和 TCR 能够更好地学习和利用时空上下文。从表 VI 中我们还发现，从传播特征中学习时空上下文的改进幅度大于从重建帧中学习时空上下文的改进幅度。这表明与 重建帧相比，该特征可能包含更多的时间信息。

*2) 时空语境对不同成分的影响：*在时态上下文重新填充过程中，我们将时态上下文输入上下文编码器-解码器、帧生成器和时态上下文编码器，以提高组合性能。我们进行了一项消融研究，以探索时空语境对不同组件的影响。具体来说，我们分别删除了编码器、解码器、帧生成器和熵模型中的时态上下文。表 VII 显示了四种变体的 BD 速率。基线是我们的最终解决方案。我们可以看到，重新填充的时空上下文在时空预测、帧重构和时空熵建模中发挥了重要作用。

*3) 中药层级数和重新填充时空语境的影响：*为了研究 TCM 模块层级数的影响，我们改变了 TCM 模块的层级数。

表九
简单增加网络层的效果。

	B	C	D	E	RGB
3L3C	0.0	0.0	0.0	0.0	0.0
1L1C	9.4	11.0	12.1	5.0	3.6
1L1C (fe)	7.7	9.0	9.9	4.3	3.0
1L1C（学分）	8.1	8.2	8.9	4.9	2.9
1L1C (fg)	8.3	9.5	9.7	6.9	3.5

表十
传播特征尺寸的影响。

	B	C	D	E	RGB
64 个通道	0.0	0.0	0.0	0.0	0.0
48 个通道	1.3	1.1	1.0	1.0	0.1
15 个频道	2.6	1.9	2.3	2.7	1.7
9 个频道	2.5	2.6	2.7	2.7	1.8

但只输出单一尺度的时间上下文 C^{-0}，例如，只有 C^{-0} 在 "TCM "中被使用。

t t

nL1C 是指中医治疗的层级数。

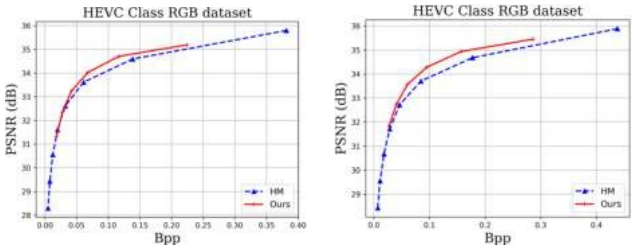
但只能输出单一比例上下文。基线是我们的最终解决方案。如表 VIII 所示，随着 TCM 模块级别的增加，比特率的节省有所改善，但当级别数为 3 时，比特率的节省开始达到饱和。

为了研究重新填充时空上下文数量的影响，我们改变了 TCM 的输出上下文数量。表 VIII 显示，随着重新填充时空上下文数量的增加，比特率节省情况有所改善。当重新填充的时空上下文数量超过 3 个时，性能提升并不明显。本文采用 3 个时空上下文。

为了证明压缩率的提高并不只是因为使用了更多的网络层，我们在公式（1）中的特征提取（*fe*）模块、公式（4）中的上下文细化（*cr*）模块和图 2 中的 1L1C 模型的帧生成器（*fg*）中添加了更多的残差块，以使澳门威尼斯人官网具与我们的最终解决方案相媲美。我们将修改后的模型分别称为 *I*L1C(*fe*)、*I*L1C(*cr*) 和 *I*L1C(*fg*)。表 IX 显示，仅仅增加网络层并不能大大提高压缩率。

*4) 传播特征维度的影响：*为了进行公平比较，我们选择 64 作为特征维度，使 DPB 大小与 RLVC [52] 相同。为了探索传播特征维度的影响，我们还将特征维度改为 48（与 FVC [22] 的 DPB 大小相同）、15（与 MLVC [24] 的 DPB 大小相同）和 9（与 4 个参考帧编解码器的 DPB 大小相同）。表 X 显示，低维度只会带来轻微的损失。用户可根据硬件能力自适应调整维数。

*5) 更宽的比特率范围：*在以前学习的视频编解码器中，通常会用不同的比特率来训练四个模型。为了便于与以往学习的视频编解码器进行比较，我们提出的方案采用了相同的设置。不过，我们仍有必要探讨所学视频编解码器在带宽受限（较低比特率）和宽带（较高比特率）环境下是否表现良好。因此，在面向 PSNR 时，我们将 λ 设为 128 和 4096，以训练



期间内 32 期间内 12

图 12.当比特率扩展到更低和更高时，我们提出的方案在 HEVC RGB 类数据集上的速率-失真性能。

额外的两个模型。在面向 MS-SSIM 时，我们将 λ 设为 4 和 128。对于其他已学习的视频编解码器（DVC_ Pro [4]、MLVC [9]、RLVC [16]、DCVC [27]），由于其正式发布的模型仅支持四个比特率点，我们无法扩展其比特率范围。图 12 展示了我们提出的方案在 HEVC 上的速率-失真性能。
比特率扩展到更低和更高时的 RGB 级。

结果表明，在更大的比特率范围内，我们的方案仍然可以运行良好。

V. 结论与讨论

在本文中，我们提出了一种时态上下文挖掘模块和时态上下文重新填充程序，以更好地学习和利用时态上下文进行学习视频压缩。在不使用自动回归熵模型的情况下，我们提出的方案比现有的学习视频编解码器实现了更高的压缩率。在 PSNR 方面，我们的方案比 H.265/HEVC-HM 的参考软件高出 14.4%；在 MS-SSIM 方面，我们的方案比 H.266/VVC- VTM 的参考软件高出 21.1%。即使就 MS-SSIM 而言，提议的方案在低延迟编码的最高压缩比设置下优于 VTM，但就 PSNR 而言，却远远落后于 VTM。此外，该方案的复杂度，尤其是解码复杂度仍需大幅降低，才能投入实际应用。我们将在今后的工作中进一步研究。

参考资料

[1] T.Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp.

[2] G. J. Sullivan、J.-R.Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp.

[3] B.Bross, Y.-K. Wang, Y. Ye.Wang、Y. Ye、S. Liu、J. Chen、G. J. Sullivan 和 J. R.Ohm，"多功能视频编码（VVC）标准及其应用概述"，《*电气和电子工程师学会视频电路与系统技术论文集》（IEEE Transactions on Circuits and Systems for Video Technology*），2021 年。

[4] G. Lu, X. Zhang, W. Ouyang, L. Chen, Z. Gao, and D. Xu, "An end-to-end learning framework for video compression," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[5] O.Rippel, A. G. Anderson, K. Tatwawadi, S. Nair, C. Lytle, and L.Bourdev, "ELF-VC: Efficient learned flexible-rate video coding," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp.

[6] H.Liu, H. Shen, L. Huang, M. Lu, T. Chen, and Z. Ma, "Learned video compression via joint spatial-temporal correlation exploration," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34、pp.11580-11587, 2020.

[7] Z.Hu, Z. Chen, D. Xu, G. Lu, W. Ouyang, and S. Gu, "Improving deep video compression by resolution-adaptive flow coding," in *European Conference on Computer Vision (ECCV)*, pp.

[8] G. Lu, C. Cai, X. Zhang, L. Chen, W. Ouyang, D. Xu, and Z. Gao, "Content adaptive and error propagation aware deep video compression," in *European Conference on Computer Vision (ECCV)*, pp.

[9] J.Lin, D. Liu, H. Li, and F. Wu, "M-LVC: multiple frames prediction for learned video compression," in *Proceedings of the IEEE/CVF* Con- *ference on Computer Vision and Pattern Recognition (CVPR)*, pp.

[10] Z.Hu, G. Lu, and D. Xu, "FVC: A new framework towards deep video compression in feature space," in *Proceedings of the IEEE/CVF Confer- ence on Computer Vision and Pattern Recognition (CVPR)*, pp.

[11] R.Yang, F. Mentzer, L. V. Gool, and R. Timofte, "Learning for video compression with hierarchical quality and recurrent enhancement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.

[12] E.Agustsson, D. Minnen, N. Johnston, J. Balle, S. J. Hwang, and G. Toderici, "Scale-space flow for end-to-end optimized video compres- sion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.

[13] Z.Cheng, H. Sun, M. Takeuchi, and J. Katto, "Learning image and video compression through spatial-temporal energy compaction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.

[14] O.Rippel, S. Nair, C. Lew, S. Branson, A. G. Anderson, and L. Bour- dev, "Learned video compression," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp.

[15] A.Djelouah, J. Campos, S. Schaub-Meyer, and C. Schroers, "Neu- ral inter-frame compression for video coding," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*、pp.6421-6429, 2019.

[16] R.Yang, F. Mentzer, L. Van Gool, and R. Timofte, "Learning for video compression with recurrent auto-encoder and recurrent probability model," *IEEE Journal of Selected Topics in Signal Processing*, vol. 15, no. 2, pp.

[17] C.-Y. Wu、N.Wu, N. Singhal, and P. Krahenbuhl, "Video compression through image interpolation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp.

[18] B.Liu, Y. Chen, S. Liu, and H.-S.Kim, "Deep learning in latent space for video prediction and compression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.

[19] H.Liu, M. Lu, Z. Ma, F. Wang, Z. Xie, X. Cao, and Y. Wang, "Neural video coding using multiscale motion compensation and spatiotemporal context model," *IEEE Transactions on Circuits and Systems for Video Technology*, 2020.

[20] M.A. Yılmaz and A. M. Tekalp, "End-to-end rate-distortion optimized learned hierarchical bi-directional video compression," *IEEE Transac- tions on Image Processing*, vol. 31, pp.

[21] Z.Chen, T. He, X. Jin, and F. Wu, "Learning for video compression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 2, pp.

[22] S.Ma, X. Zhang, C. Jia, Z. Zhao, S. Wang, and S. Wang, "Image and Video compression with neural networks： A review," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 6, pp.

[23] A.Habibian, T. v. Rozendaal, J. M. Tomczak, and T. S. Cohen, "Video compression with rate-distortion autoencoders," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[24] W.Sun, C. Tang, W. Li, Z. Yuan, H. Yang, and Y. Liu, "High-quality single-model deep video compression with frame-conv3d and multi- frame differential modulation," in *European Conference on Computer Vision (ECCV)*, pp.

[25] J.Pessoa, H. Aidos, P. Toma´s, and M. A. Figueiredo, "End-to-end learning of video compression using spatio-temporal autoencoders," in 2020 *IEEE Workshop on Signal Processing Systems (SiPS)*, pp.

[26] J.Liu, S. Wang, W.-C. Ma, M. Shah, R. Hu, P. Dhawan, and R. Urtasun, "Conditional entropy coding for efficient video compression," in Euro- pean ConferenceMa, M. Shah, R. Hu, P. Dhawan, and R. Urtasun, "Conditional entropy coding for efficient video compression," in *Euro- pean Conference on Computer Vision (ECCV)*, pp.

[27] J.Li, B. Li, and Y. Lu, "Deep contextual video compression," *Advances in Neural Information Processing Systems*, vol. 34, pp.

[28] J.Wang, X. Deng, M. Xu, C. Chen, and Y. Song, "Multi-level wavelet- based generative adversarial network for perceptual quality enhancement of compressed video," in *European Conference on Computer Vision (ECCV)*, pp.

[29] R.Yang, R. Timofte, and L. Van Gool, "Perceptual learned video compression with recurrent conditional gan," in *Proceedings of the Thirty-First International Joint Conference on Artificial* Intelligence*, IJCAI-22*, pp.

[30] W.Ding, F. Wu, X. Wu, S. Li, and H. Li, "Adaptive directional lifting- based wavelet transform for image coding," *IEEE Transactions on Image Processing*, vol. 16, no.

[31] F.Wu, S. Li, and Y.-Q.Zhang, "A framework for efficient progressive fine granularity scalable video coding," *IEEE Transactions on Circuits and Systems for Video Technology.*

[32] H.Ma, D. Liu, N. Yan, H. Li, and F. Wu, "End-to-end optimized versatile image compression with wavelet-like transform," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[33] "x265。" https://www.videolan.org/developers/x265.html。访问： 2022-07-05。

[34] "JM-19.0。" http://iphome.hhi.de/suehring/。访问日期：2022-07-05。

[35] "HM-16.20." https://vcgit.hhi.fraunhofer.de/jvet/HM/。访问时间： 2022-07-05.

[36] "vtm-13.2." https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware vtm/。访问日期：2022-03-02。

[37] R.Yang, Y. Yang, J. Marino, and S. Mandt, "Hierarchical autoregressive modeling for neural video compression," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021,* OpenReview.net, 2021.

[38] D.Minnen, J. Balle´, and G. Toderici, "Joint autoregressive and hierar- chical priors for learned image compression," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8*、*2018年，加拿大蒙特里尔*，第10794-10803页，2018年。

[39] Z.Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, vol. 2, pp.

[40] Z.Guo, Z. Zhang, R. Feng, and Z. Chen, "Causal contextual prediction for learned image compression," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[41] Z.Cheng, H. Sun, M. Takeuchi, and J. Katto, "Learned image com- pression with discretized gaussian mixture likelihoods and attention modules," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[42] "x264。" https://www.videolan.org/developers/x264.html。访问： 2022-07-05。

[43] F.Mentzer, E. Agustsson, J. Balle´, D. Minnen, N. Johnston, and G. Toderici, "Neural video compression using gans for detail synthesis and propagation," in *European Conference on Computer Vision*, pp.

[44] A.Ranjan 和 M. J. Black，"使用空间金字塔网络进行光流估计"，《*IEEE/CVF 计算机视觉与模式识别会议（CVPR）论文集*》，第 4161-4170 页， 2017。

[45] J.Balle´, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," in *6th International Confer- ence on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, OpenReview.net, 2018.

[46] J.Be´gaint, F. Racape´, S. Feltman, and A. Pushparaja, "Compressai: a pytorch library and evaluation platform for end-to-end compression research," *arXiv preprint arXiv:2011.03029*, 2020.

[47] R.Feng, Y. Wu, Z. Guo, Z. Zhang, and Z. Chen, "Learned video com- pression with feature-level residuals," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*、pp.120-121, 2020.

[48] K.He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image *recognition*," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.

[49] W.Shi, J. Caballero, F. Husza´r, J. Totz, A. P. Aitken, R. Bishop、D. Rueckert 和 Z. Wang，"利用高效的亚像素卷积神经网络实现单幅图像和视频的实时超分辨率"。

in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.

[50] K.Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser：用于图像去噪的深度 cnn 的残差学习》，*IEEE Transactions on Image Processing*，第 26 卷，第 7 期，第 3142-3155 页，2017 年。

[51] Z.Guo, R. Feng, Z. Zhang, X. Jin, and Z. Chen, "Learning cross scale prediction for efficient neural video compression," *arXiv preprint arXiv:2112.13309*, 2021.

[52] D.Flynn, D. Marpe, M. Naccari, T. Nguyen, C. Rosewarne, K. Sharman、J.Sole, and J. Xu, "Overview of the range extensions for the hevc standard：工具、配置文件和性能"，《*IEEE 视频技术电路与系统论文集*》，第 26 卷，第 1 期，第 4-19 页， 2015。

[53] D.Ma, F. Zhang, and D. Bull, "BVI-DVC: A training database for deep video compression," *IEEE Transactions on Multimedia*, 2021.

[54] A.V. Katsenou、G. Dimitrov、D. Ma 和 D. R. Bull，"Bvi-syntex：A synthetic video texture d a t a s e t for video compression and quality assessment," *IEEE Transactions on Multimedia*, vol. 23, pp.

[55] T.Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman, "Video enhance- ment with task-oriented flow," *International Journal of Computer Vision*, vol. 127, no. 8, pp.

[56] A.Mercat, M. Viitanen, and J. Vanne, "UVG dataset：用于视频编解码器分析和开发的 50/120fps 4k 序列"，《*第 11 届 ACM 多媒体系统会议论文集*》，第 297-302 页，2020 年。

[57] H.Wang, W. Gan, S. Hu, J. Y. Lin, L. Jin, L. Song, P. Wang、I.I. Katsavounidis, A. Aaron, and C.-C.J. Kuo, "MCL-JCV: a JND- based H.264/AVC video quality assessment dataset," in *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 1509-1513, IEEE, 2016.

[58] I.Loshchilov 和 F. Hutter："去耦合权重衰减正则化"，*第 7 届学习表征国际会议，ICLR 2019， New Orleans, LA, USA, May 6-9, 2019,* OpenReview.net，2019。

[59] G. Bjontegaard，"计算 rd- 曲线之间的平均 psnr 差异"，*VCEG-M33*，2001 年。

盛希华于 2019年获得中国沈阳东北大学自动化专业学士学位。他目前在合肥中国科学技术大学电子工程与信息科学系攻读博士学位。他的研究兴趣包括图像/视频/点云编码、信号处理和机器学习。

李家豪于2014年获得哈尔滨工业大学计算机科学与技术学士学位，2019年获得北京大学博士学位。现任微软亚洲研究院媒体计算组高级研究员。曾从事视频编码研究，在该领域发表论文、标准提案和专利十余篇。他目前的研究兴趣主要集中在神经视频压缩和实时通信。

李斌分别于2008年和2013年获得安徽合肥中国科学技术大学电子工程学士和博士学位。他于 2013 年加入中国北京微软亚洲研究院（MSRA），现任首席研究员。他撰写或合作撰写了50多篇论文。他在图像和视频编码领域拥有30多项已获授权或正在申请的美国专利。他有 40 多项技术提案被视频编码联合协作组采纳。他的目前的研究兴趣包括视频编码、处理、传输和通信。

李博士于 2011 年获得美国计算机协会颁发的移动和泛在多媒体国际会议最佳论文奖。他获得了2014年IEEE国际图像处理大会的前10%论文奖。曾获2017年IEEE视觉通信与图像处理最佳论文奖。

李力（男，17岁）分别于2011年和2016年获得安徽合肥中国科学技术大学电子工程学士和博士学位。2016年至2020年，他在美国密苏里大学堪萨斯城分校担任客座助理教授。他于 2020 年加入中国科学技术大学电子工程与信息科学系，担任研究员，并于 2022 年成为教授。

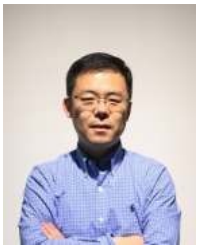
他的研究兴趣包括图像/视频/点云编码和处理。他曾获得最佳2016 年 IEEE 视觉通信与图像处理（VCIP）和 2019 年 IEEE 图像处理国际会议（ICIP）10% 论文奖。

刘东（M'13-SM'19）分别于 2004 年和 2009 年获得中国科学技术大学（合肥）电子工程学士和博士学位。2009 年至 2012 年，他是 中国北京诺基亚研究中心的研究人员。2012 年加入中国科学技术大学任教，2020 年晋升为教授。

他的研究兴趣包括图像和视频处理、编码、分析和数据挖掘。他撰写或合作撰写了 200 多篇论文在国际期刊和会议上发表的论文。他拥有 20 多项授权专利。他的多项技术提案被国际或国内标准化组织采纳。他曾获得 2009 年电气和电子工程师学会视频技术电路与系统 TRANSACTIONS 最佳论文奖，以及 2009 年电气和电子工程师学会视频技术研讨会优秀论文奖。

VCIP 2016 最佳 10% 论文奖。他和他的学生在 ISCAS 2022、ICCV 2019、ACM MM 2019、ACM MM 2018、ECCV 2018、CVPR 2018 和 ICME 2016 举行的多项技术挑战赛中获奖。他是

CCF和CSIG高级会员，IEEE CAS学会MSA-TC当选会员。他担任或曾担任 IEEE 1857.11 标准工作组主席，IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY 的客座编辑，*Frontiers in Signal Processing* 的副主编，VCIP 2022、ICME 2021、ICME 2019 等会议的组织委员会成员。



Yan Lu 在中国哈尔滨工业大学获得计算机科学博士学位。他 于 2004 年加入微软亚洲研究院，现任合作伙伴研究经理，负责管理媒体计算和通信方面的研究。他和他的团队将许多关键技术和研究原型转化为微软产品。2001 年至 2004 年，他担任中国计算技术研究所 JDL 实验室视频编码组组长。1999 年至 2000 年，他在香港城市大学担任研究员。

助理。Yan Lu 在实时通 信 、计算机视觉、视频分析、音频增强、虚拟化和移动云计算领域有着广泛的研究兴趣。他拥有 30 多项已获授权的美国专利，并在权威期刊和会议论文集上发表了 100 多篇论文。