2019 IEEE/CVF 计算机视觉与模式识别大会（CVPR）

DVC：端到端深度视频压缩框架

Guo Lu[1], Wanli Ouyang[2], Dong Xu[3], Xiaoyun Zhang[1], Chunlei Cai[(1)], and Zhiyong Gao*[1] [1]Shanghai Jiao Tong University, {luguo2014, xiaoyun.zhang, caichunlei, zhiyong.gao}@sjtu.edu.cn

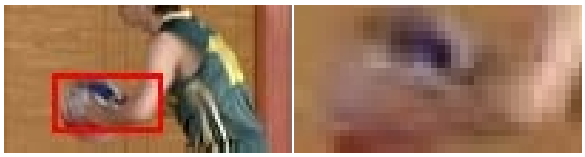
[2]澳大利亚悉尼大学，SenseTime 计算机视觉研究小组 [3]悉尼大学，{wanli.ouyang, dong.xu}@sydney.edu.au

摘要

传统的视频压缩方法采用预判定编码架构，并对相应的运动信息和残差信息进行编码。在本文中利用传统视频压缩方法中的经典架构和神经网络强大的非线性表示能力，我们提出了首个端到端视频压缩深度模型，该模型可联合优化视频压缩的所有组件。具体来说，我们利用基于学习的光流估计来获取运动信息并重建当前帧。然后，我们采用两个自动编码器式神经网络来压缩相应的运动和残留信息。所有模块都是通过一个单一的损失函数来共同学习的，其中它们通过考虑减少压缩比特数和提高解码视频质量之间的权衡来相互协作。实验结果表明，所提出的方法在 PSNR 方面优于广泛使用的视频编码标准 H.264，在 MS-SSIM 方面甚至与最新标准 H.265 不相上下。代码发布于 https://github.com/GuoLusjtu/DVC。

1. 导言

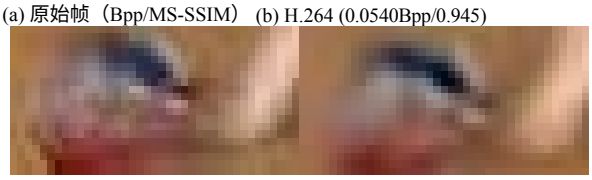
如今，视频内容占互联网流量的 80% 以上 [26]，预计这一比例还会进一步增加。因此，建立高效的视频压缩系统并在给定的带宽预算内生成更的帧至关重要。此外，大多数与视频相关的计算机视觉任务（如视频对象检测或视频对象跟踪）对压缩视频质量都很敏感，高效的视频压缩可能会为其他计算机视觉任务带来好处。同时，视频压缩技术也有助于动作识别 [41] 和模型压缩 [16]。



然而，在过去的几十年中，视频压缩算法 [39, 31] 都依赖于手工制作的模块，如基于块的运动估计和离散余弦变换 (DCT)，以减少视频序列中的冗余。

*通讯作者

(a) 原始帧（Bpp/MS-SSIM） (b) H.264 (0.0540Bpp/0.945)



(c) H.265 (0.082Bpp/0.960) (d) Ours (**0.**0529Bpp/ **0.961**)

图 1：不同视频压缩系统重建帧的视觉质量。(a) 为原始帧。(b)-(d) 为 H.264 的重建帧、H.265 和我们的方法。通过 MS-SSIM 测量，我们提出的方法只需 0.0529pp，而感知质量（0.961）却是最好的。（最佳观看效果为彩色）。

虽然每个模块都经过精心设计，但整个压缩系统并没有进行端到端的优化。我们希望通过联合优化整个压缩系统，进一步提高视频压缩性能。

最近，基于深度神经网络（DNN）的自动图像压缩编码器[34, 11, 35, 8, 12, 19, 33, 21, 28, 9]取得了与 JPEG [37], JPEG2000 [29] 或 BPG [1] 等传统图像编解码器相当甚至更好的性能。一种可能的解释是，基于 DNN 的图像压缩方法可以利用大规模端到端训练和高度非线性变换，而这些在传统方法中都没有使用。

然而，直接应用这些技术来构建端到端视频压缩学习系统并非易事。**首先**，学习如何生成和压缩用于视频压缩的运动信息仍是一个未决问题。视频压缩方法在很大程度上依赖于运动信息来减少视频序列中的时间冗余。一个直接的解决方案是使用基于学习的光流来表示运动信息。然而，目前基于学习的光流方法旨在生成尽可能精确的流场。但是，精确的光流往往不是最优的。

978-1-7281-3293-8/19/$31.00 ©2019 IEEE
doi 10.1109/cvpr.2019.01126

对于一项特定的视频任务而言，光学流是非常有害的[42]。此外，与传统压缩系统中的运动信息相比，光流的数据量大幅增加，直接应用 [39, 31] 中现有的压缩方法来压缩光流值将大幅增加存储运动信息所需的比特数。**其次**，目前还不清楚如何通过最小化基于残差和运动信息的速率失真目标来构建基于 DNN 的视频压缩系统。速率失真优化（RDO）的目的是在给定用于组合压缩的比特数（或比特率）时，获得更高质量的重构帧（即更小的失真）。RDO 对视频压缩性能非常重要。为了在基于学习的压缩系统中发挥端到端训练的威力，需要采用 RDO 策略来优化整个系统。

在中，我们提出了首个端到端深度视频压缩（DVC）模型，该模型可联合学习运动定时、运动压缩和残差压缩。该网络的优势可归纳如下：

- 视频压缩的所有关键部分，*即*运动估计、运动补偿、残差压缩、运动压缩、量化和比特率估计，都是通过端到端神经网络实现的。
- 通过单一损失函数，基于速率-失真权衡对视频压缩的关键部分进行联合优化，从而提高压缩效率。
- 传统视频压缩方法的组成部分与我们提出的 DVC 模型之间存在一一对应的关系。这项工作为从事视频压缩、计算机视觉和深度模型设计的研究人员架起了一座桥梁。例如，用于光流估计和图像组合的更好的模型可以很容易地插入到我们的框架中。这些领域的研究人员可以把我们的 DVC 模型作为他们未来研究的起点。

实验结果表明，使用我们基于神经网络的方法估计和压缩运动信息，可以显著提高压缩性能。根据 PSNR 测量，我们的框架优于广泛使用的视频编解码器 H.264；根据多尺度结构相似性指数（MS-SSIM）测量，我们的框架与最新的视频编解码器 H.265 相当[38]。

2. 相关工作

2.1. 图像压缩

在过去的几十年里，人们提出了许多图像压缩算法 [37, 29, 1]。这些方法

通常依赖于手工技术。，JPEG 标准通过使用 DCT 将像素线性映射到另一种表示形式，并在进行熵编码之前量化相应的系数[37]。缺点之一是这些模块需要单独优化，可能无法达到最佳压缩性能。

最近，基于 DNN 的图像压缩方法引起了越来越多的关注 [34, 35, 11, 12, 33, 8]、
21, 28, 24, 9].在 [34、35、19] 中，利用递归神经网络 (RNN) 建立了渐进式图像压缩方案。其他方法则采用 CNN 来构建自动编码器式网络，用于图像压缩[11, 12, 33]。为了优化神经网络，[34、35、19] 中的工作只试图最小化原始帧和重新构建帧之间的失真（*如*均方误差），而没有考虑用于压缩的比特数。为了提高压缩效率，[11、12、33、21] 采用了速率-失真优化技术，在优化过程中引入比特数。为了估算比特率，[28, 21, 24]中的自适应算术编码方法学习了上下文模，而[11, 33]中则使用了非自适应算术编码。此外，其他技术，如广义除法归一化（GDN）[11]、多尺度图像分解[28]、对抗训练[28]、重要度[33][34].....
图 [21, 24] 和内部预测 [25, 10]，以提高图像压缩性能。这些已有的研究成果是我们的视频压缩网络的重要组成部分。

2.2. 视频压缩

在过去几十年中，人们提出了几种传统的视频压缩算法，如 H.264 [39] 和 H.265 [40]。
H.265 [31].这些算法大多采用预测编码架构。虽然它们能提供高效的压缩性能，但它们都是人工设计的，无法以端到端的方式进行联合优化。

针对视频压缩任务，人们提出了许多基于 DNN 的方法，用于内部预测和残差编码[13]、模式决策[22]、熵编码[30]和后处理[23]。这些方法用于提高传统视频压缩算法中某一特定模块的性能，而不是构建端到端的压缩方案。在 [14] 中，Chen *等人*提出了一种基于块的视频压缩学习方法。然而，这种方法不可避免地会在块与块之间的边界产生块状伪影。此外，他们通过传统的基于块的运动估计，使用了从以前的重建帧中传播的运动信息，这将降低压缩性能。Tsai *等人*提出了一种自动编码器网络，用于压缩特定领域视频的 H.264 编码器残差[36]。这项工作没有使用深度模型进行运动估计、

运动补偿或运动压缩

最相关的工作是 [40] 中基于 RNN 的方法，其中视频压缩被表述为帧内插值。然而，在他们的方法中，运动信息也是由传统的基于块的运动估计产生的，并由现有的基于非深度学习的图像压缩方法进行编码[5]。换句话说，运动的估计和压缩不是由深度模型完成的，而是与其他组件共同优化的。此外，[40] 中的视频编解码器只着眼于最小化原始帧与重建帧之间的失真（*即*均方误差），而没有在训练过程中考虑速率与失真之间的权衡。相比之下，在我们的网络中，运动估计和压缩是通过 DNN 实现的，而 DNN 是在考虑整个压缩系统的速率-失真权衡的基础上与其他组件共同优化的。

2.3. 运动估计

运动估计是视频通信系统的一个关键组成部分。传统的视频编解码器使用基于块的运动估计算法 [39]，这种算法非常适合硬件实现。

在计算机视觉任务中，光流被广泛用于利用时间关系。最近，人们提出了很多基于学习的光流估计方法 [15, 27, 32, 17, 18]。这些方法促使我们光流估计纳入端到端学习框架。与现有视频压缩中基于块的运动估计方法相比，基于学习的光流方法可以提供像素级的精确运动信息，而且还能以端到端方式进行操作。但是，如果采用传统的视频压缩方法对光流值进行编码，则需要更多比特来压缩运动信息。

3. 建议的方法

导言 of Notations. 让 $V = \{x_1, x_2, \ldots, x_{(t)-1}, x_t, \ldots\}$ 表示当前视频序列，其中 x_t 是时间步长为 t 的帧。预测帧表示为 $\bar{x}_{(t)}$，重建/解码帧表示为 $\hat{x}_t$。r_t 表示原始帧 $x_{(t)}$ 和预测帧之间的残差（误差）。

$\bar{x}_t$。$\hat{r}_t$ 表示重建/解码后的残差。在

为了减少时间冗余，运动信息。，v_t 表示运动矢量或光流值。而 $\hat{v}_t$ 则是其相应的重构版本。可以使用线性或非线性变换来提高压缩效率。因此，残差信息 r_t 可以

3.1. 视频压缩简介

在本节中，我们将简要介绍视频通信。更多详情请参见文献 [39, 31]。一般来说，视频压缩编码器根据输入的当前帧生成比特流。解码器根据接收到的比特流重新构造视频帧。在图 2 中，编码器端包含所有模块，而解码器端不包含蓝色模块。

图 2(a) 所示的经典视频压缩框架采用预测变换架构。具体来说，输入帧 $x_{(t)}$ 被分割成一组大小相同的块，*即*正方形区域（*如* 8× 8）。传统视频压缩算法在编码器端的编码过程如下、

步骤 1.运动估计。估计当前帧 x_t 和上一重建帧 $\hat{x}_{t-1}$ 之间的运动。得到每个块的相应运动向量 v_t。

步骤 2.运动补偿。根据步骤 1 中定义的运动矢量 v_t，将之前重建帧中的相应像素复制到当前帧中，得到预测帧 $\bar{x}_{(t)}$。原始帧 $x_{(t)}$ 和预测帧 $\bar{x}_t$ 之间的残差 r_t 为 $r_t = x_{(t)} - \bar{x}_{(t)}$。

步骤 3.变换和量化。将步骤 2 中的残差 r_t 量化为 $\hat{y}_t$。量化前使用线性变换（*如* DCT）以获得更好的压缩性能。

步骤 4.反变换。步骤 3 中的量化结果 $\hat{y}_t$ 通过逆变换得到重组后的残差 $\hat{r}_{(t)}$。

步骤 5.熵编码。步骤 1 中的运动矢量 v_t 和步骤 3 中的量化结果 $\hat{y}_t$ 都将通过熵编码方法编码成比特，并发送给解码器。

步骤 6.帧重构。将步骤 2 中的 $\bar{x}_t$ 和步骤 4 中的 $\hat{r}_{(t)}$ 相加，*即*得到重建帧 $x_{(t)}$，*即* $\hat{x}_t = \hat{r}_{(t)} + \bar{x}_t$。重建后的帧将被步骤 1 中的第（$t+1$）帧用于运动估计。

对于解码器来说，根据编码器在步骤 5 中提供的比特，在步骤 2 中进行运动补偿，在步骤 4 中进行反量化，然后在步骤 6 中进行帧重构，以获得重构帧 $\hat{x}_t$。

3.2. 拟议方法概述

转换为 $y_{(t)}$，运动信息 $v_{(t)}$ 可以转换为 $m_{(t)}$，$\hat{r}_{(t)}$ 和 $\hat{m}_t$ 分别是相应的量化版本。

图 2 (b) 是我们端到端视频压缩框架的高级概览。传统视频压缩框架与我们提出的基于深度学习的框架之间存在一一对应的关系。两者的关系和不同之处简要总结如下：

步骤 N1.运动估计和压缩。我们使用 CNN 模型来估计光流[27]，即

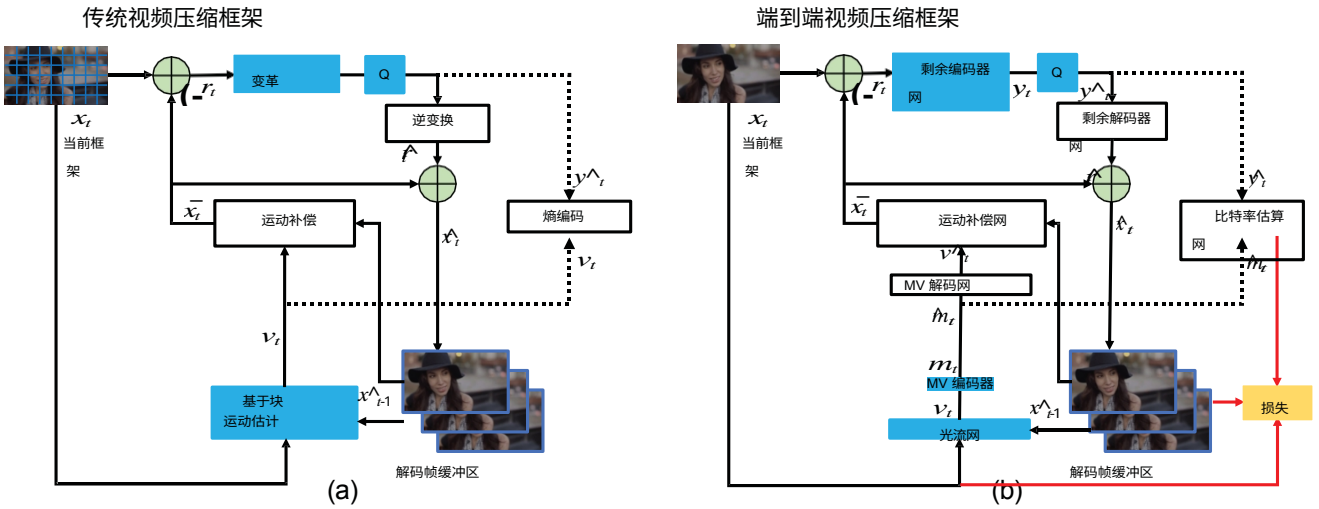


图 2：(a)：传统视频编解码器 H.264 [39] 或 H.265 [31] 使用的预测编码架构。(b): 拟议的端到端视频压缩网络。蓝色的模块不包括在解码器中。

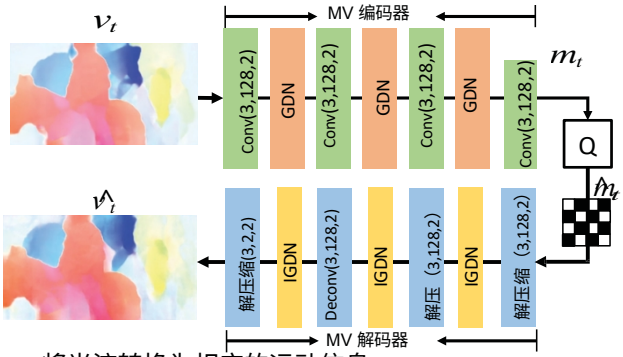
被视为运动信息 v_t。图 2 中提出了一个 MV 编码器-解码器网络来压缩和解码光流值，而不是直接对原始光流值进行编码。然后使用 MV 解码网对相应的重新结构化运动信息 $\hat{v}_t$ 进行解码。详情见第 3.3 节。

步骤 N2。运动补偿。根据步骤 N1 获得的光流，设计一个运动补偿网络来获得预测帧 $\bar{x}_{(t)}$。更多信息请参见第 3.4 节。

步骤 N3-N4。变换、量化和反变换。我们使用高度非线性的残差编码器-解码器网络来取代步骤 3 中的线性变换，并残差 r_t 非线性地映射到表示数 $y_{(t)}$。然后将 y_t 量化为 $\hat{y}_t$。为了建立端到端训练方案，我们使用了 [11] 中的量化方法。将量化后的 $\hat{y}_t$ 输入残差解码器网络，得到重构后的残差 $\hat{r}_t$。详情见第 3.5 和 3.6 节。

步骤 N5.熵编码。在测试阶段，来自步骤 N1 的量化运动表示 $\hat{m}_{(t)}$ 和来自步骤 N3 的残差表示 $\hat{y}_{(t)}$ 被编码成比特，并发送给解码器。在训练阶段，为了估算我们提出的方法所需的比特数，我们使用 CNN（图 2 中的比特率估算网）来获取 $\hat{m}_t$ 和 $\hat{y}_t$ 中每个符号的概率分布。更多信息请参见第 3.6 节。

步骤 N6。帧重构。与第 3.1 节中的步骤 6 相同。



，将光流转换为相应的运动信息。

3.3. MV 编码器和解码器网络

为了在步骤 N1 压缩运动信息，我们设计了一个 CNN

图 3：我们的 MV 编码器-解码器网络。Conv(3,128,2)表示卷积操作，内核大小为 3x3，输出通道为 128，跨距为 2。 GDN/IGDN [11] 是非线性变换函数。二进制特征图仅用于说明。

我们利用自动编码器风格的网络来压缩光流，以获得更好的编码效果。具体来说，我们利用自动编码器风格的网络来压缩光流，这是由文献[11]首次针对图像压缩任务提出的。整个 MV 压缩网络如图 3 所示。光流 v_t 被送入一系列卷积运算和非线性变换。卷积（解卷积）的输出通道数为 128 个，但最后一个解卷积层的输出通道数等于 2。给定光流 v_t 的大小为 $M\times N\times 2$，MV 编码器将生成运动表示 $m_{(t)}$，其大小为 M/16× N/16× 128。然后将 m_t 量化为 $\hat{m}_t$。MV 解码器接收量化表示并重建运动信息 $\hat{v}_t$。此外，量化表示 $\hat{m}_t$ 将用于熵编码。

3.4. 运动补偿网络

给定上一重建帧 $\hat{x}_{t-1}$ 和向量

根据 $\hat{v}_t$，运动补偿网络可获得预测帧 $\bar{x}_t$，该帧应尽可能接近当前帧 x_t。首先，根据运动信息 $\hat{v}_t$，将上一帧 $\hat{x}_{t-1}$ 扭曲为当前帧。扭曲后的帧仍有伪影。为了去除伪影，我们将翘曲帧 $w(\hat{x}_{t-1}, \hat{v}_t)$, 参考帧 $\hat{x}_{t-1}$ 和运动矢量 $\hat{v}_t$ 作为输入，然后将它们输入到另一个 CNN 中，以获得重新生成的预测帧 $\bar{x}_t$。拟议网络的整体架构如图 4 所示。图 4 中 CNN 的细节见补充材料。我们提出的方法是一种像素化的运动补偿方法，可以提供更准确的时间信息，并避免传统的基于块的运动补偿方法中的块状伪影。这意味着我们不需要手工制作的环路滤波器或采样自适应偏移技术 [39, 31]来进行后期处理。

3.5. 剩余编码器和解码器网络

原始帧 x_t和预测帧 $\bar{x}_t$ 之间的残差信息 r_t由残差编码器网络进行编码，如图 2 所示。在本文中，我们依靠 [12] 中的高度非线性神经网络将残差信息转换为相应的潜在表示。与传统视频压缩系统中的离散余弦变换相比，我们的方法能更好地发挥非线性变换的威力，实现更高的压缩效率。

3.6. 培训战略

损失函数我们的视频压缩框架的目标是最大限度地减少用于视频编码的比特数，同时减少原始输入帧 x_t 和重组帧 $\hat{x}_t$ 之间的失真。因此，我们提出了以下速率-失真优化问题、

$$\lambda D + R = \lambda d(x_t, \hat{x}_t) + (H(\hat{m}_t) + H(\hat{y}_t)), \quad (1)$$

其中，$d(x_t, \hat{x}_t)$ 表示 x_t 和 $\hat{x}_t$ 而我们在实现过程中使用的是均方误差 (MSE)。$H(\cdot)$ 表示用于编码表示的比特数。在我们的方法中，残差表示 $\hat{y}_t$和运动表示 $\hat{m}_t$都应编码到比特流中。如图 2(b)所示，重构帧 $\hat{x}_t$、原始帧 x_t 和估计比特输入到损耗函数。

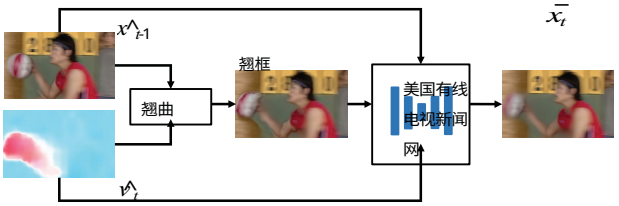


图 4：我们的运动补偿网络。

在进行熵编码之前，需要对其进行量化。然而，量化操作不具有差分性，这使得端到端训练无法实现。为了解决这个问题，人们提出了很多方法 [34, 8, 11]。本文采用 [11] 中的方法，在训练阶段加入均匀噪声来代替量化操作。以 y_t 为例，训练阶段量化表示 $\hat{y}_t$通过添加均匀噪声来近似，即 $\hat{y}_t = y_t + \eta$，其中 η 为均匀噪声。在舍入阶段，我们直接使用舍入运算，即$\hat{y}_t = round(y_t)$。

比特率估算。为了优化整个网络的比特数和失真度，我们需要获得生成的潜在表示（$\hat{y}_t$ 和 $\hat{m}_t$）的比特率（$H(\hat{y}_t)$和 $H(\hat{m}_t)$）。比特率的正确测量方法是相应的潜在表示符号的熵。因此，我们可以估计 $\hat{y}_t$和 $\hat{m}_t$的概率分布，然后得到相应的熵。本文使用 [12] 中的 CNN 来估计分布。

缓冲前一帧。如图 2 所示，运动估计和运动补偿网络在计算当前帧时需要上一帧的重建帧 $\hat{x}_{t-1}$。然而，上一帧的重建帧 $\hat{x}_{t-1}$ 是我们的网络对上一帧的输出，而上一帧的输出是基于重建帧 $\hat{x}_{t-2}$ 的，以此类推。因此，在帧 x_t 的训练过程中，可能需要帧 $x_1, \ldots, x_{t-1}$，这就减少了小批量训练样本的变化，而且当 t 较大时，GPU 中可能无法存储这些样本。为了解决这个问题，我们采用了在线更新策略。具体来说，每次迭代中重建的帧 $\hat{x}_t$ 将保存在缓冲区中。在接下来的迭代中

缓冲区中的 $\hat{x}_t$ 将被用于运动估计和模拟。
在对 x_{t+1} 进行编码时，会产生干扰补偿。因此，每个

量化。需要残差表示 y_t和运动表示 m_t等潜在表示。

缓冲区中的训练样本将在一次迭代中更新。这样，我们就能在每次迭代中优化并存储一个视频片段的一帧，从而提高效率。

4. 实验

4.1. 实验装置

数据集。我们使用 Vimeo-90k 数据集[42]来训练拟议的视频压缩框架，该数据集是为评估不同的视频处理任务而重新构建的、

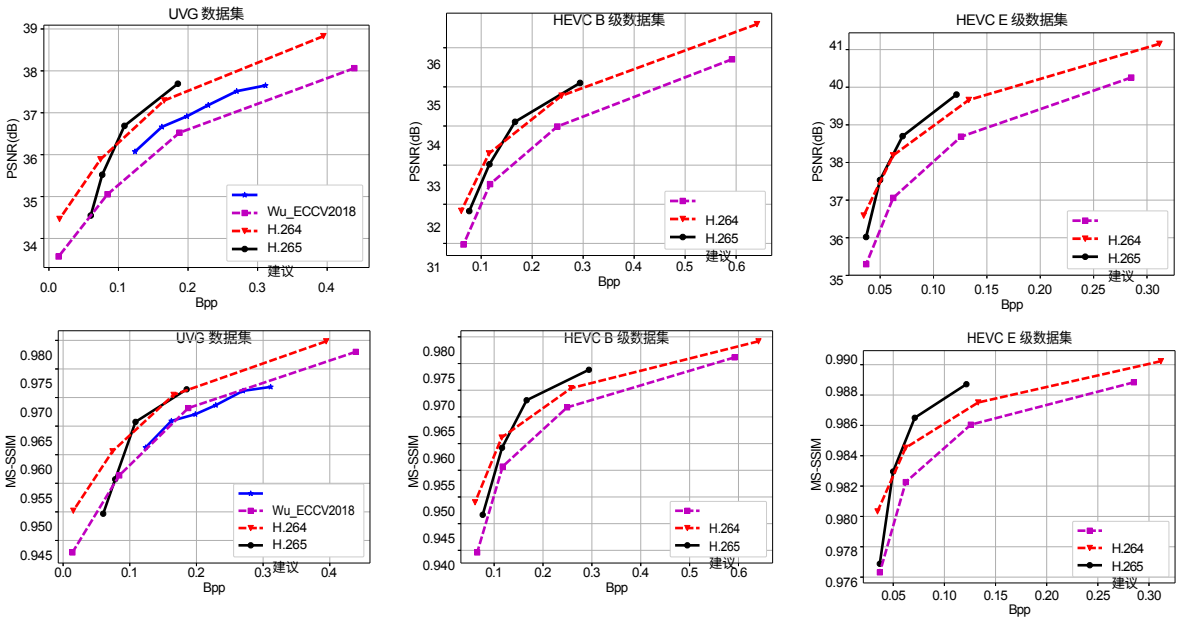


图 5：我们提出的方法与 [40]、H.264 [39] 和 H.265 [31] 中基于学习的视频编解码器的比较。根据 PSNR 和 MS-SSIM 测量，我们的方法优于 H.264。同时，就 MS-SSIM 而言，与 H.265 相比，我们的方法取得了相似或更好的压缩性能。

如视频去噪和视频超分辨率。它由 89 800 个独立的片段组成，这些片段在内容上互不相同。

为了报告我们提出的方法的性能，我们在 UVG 数据集 [4] 和 HEVC 标准测试序列（B 级、C 级、D 级和 E 级）[31] 上评估了我们提出的算法。这些数据集的内容和分辨率多种多样，被广泛用于衡量视频压缩算法的性能。**评估方法** 为了测量重新构建的帧的失真度，我们使用了两个评估指标：PSNR 和 MS-SSIM [38]。与 PSNR 相比，MS-SSIM 与人类对失真感知的相关性更好。为了衡量对图像进行编码所需的比特数，我们使用每像素比特数（Bpp）来表示每个像素所需的比特数。在当前帧中。

实现细节 我们用不同的 λ（λ= 256、512、1024、2048）训练四个模型。对于每个模型，我们使用 Adam 优化器 [20]，将初始学习率分别设置为 0.0001、β_1 0.9 和 β_2 0.999。当损失趋于稳定时，学习率除以 10。训练图像的分辨率为 256× 256。运动估计模块使用 [27] 中的预训练权重进行初始化。整个系统基于 Tensorflow 实现，使用两个 Titan X GPU 训练整个网络大约需要 7 天时间。

4.2. 实验结果

本节将 H.264 [39] 和 H.265 [31] 纳入比较范围。此外，[40] 中的基于学习的视频压缩系统（用 Wu ECCV2018 表示）也被纳入比较范围。

也包括在内，以供比较。为了生成 H.264 和 H.265 压缩帧，我们采用了以下设置
[40] 并使用 FFmpeg *快速*模式。UVG 数据集和 HEVC 数据集的 GOP 大小分别为 12 和 10。有关 H.264/H.265 设置的更多详情，请参阅补充材料。

图 5 显示了 UVG 数据集、HEVC B 类和 E 类数据集的实验结果。 图 5 显示了 UVG 数据集、HEVC B 类和 E 类数据集的实验结果。HEVC Class C 和 Class D 的结果见补充材料。很明显，我们的方法大大优于最近的视频压缩研究成果 [40]。在 UVG 数据集上，建议的方法在相同 Bpp 水平下实现了约 0.6dB 的增益。值得一提的是，我们的方法只使用了一个前参考帧，而 Wu *等人*[40]的研究则使用了双向帧预测，需要两个相邻帧。因此，通过利用多个参考帧的时间信息，可以进一步提高我们框架的压缩性能。

在大多数数据集上，根据 PSNR 和 MS-SSIM 测量，我们提出的框架都优于 H.264 标准。此外，就 MS-SSIM 而言，与 H.265 相比，我们的方法实现了相似或更好的压缩性能。如前所述，我们损失函数中的失真项是用 MSE 度量的。尽管如此，就 MS-SSIM 而言，我们的方法仍能提供合理的视觉质量。

4.3. 消融研究和模型分析

运动估计。在我们提出的方法中，我们利用了端到端训练策略的优势，并优化了运动估算。

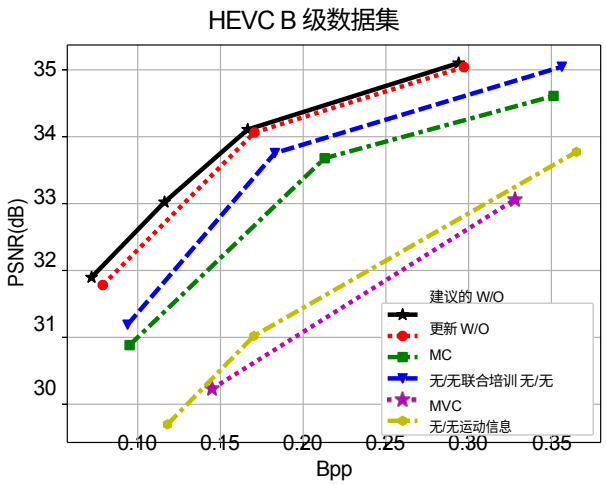


图 6：消融研究。我们报告了以下情况下的压缩性能。1. 不采用缓冲前一帧的策略（W/O 更新）。2. 移除运动补偿网络（W/O MC）。3.运动估计模块未进行联合优化（W/O Joint Training）。4.删除运动压缩网络（W/O MVC）。5. 不依赖运动信息（W/O Motion Information）。

因此，基于速率-失真优化，我们系统中的光流有望更加可压缩，从而获得更精确的扭曲帧。因此，在速率失真优化的基础上，我们系统中的光流预计会更易于压缩，从而产生更精确的扭曲帧。为了证明
我们进行了一项实验，通过固定

在整个训练阶段，对初始化运动估计模块的参数进行预训练。在这种情况下，对运动估计模块的预训练只是为了更准确地估计光流，而不是为了优化速率失真。图 6 中的实验结果表明，与图 6 中以 *"W/O 联合训练"*表示的固定运动估计方法相比，我们的运动估计联合训练方法能显著提高性能（见蓝色曲线）。

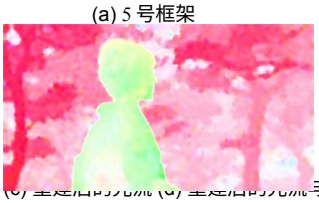
我们在表 1 中报告了光流编码的平均比特成本和扭曲帧的相应 PSNR。具体来说，当运动估计模块在训练阶段固定不变时，它需要 0.044bpp 来编码生成的光流，翘曲帧的相应 PSNR 为 27.33db。相比之下，我们提出的方法只需要 0.029bpp 就能对光流进行编码，而且翘曲帧的 PSNR 更高（28.17dB）。因此，联合学习策略不仅节省了运动编码所需的比特数，而且具有更好的扭曲图像质量。这些实验结果清楚地表明，将运动估计纳入速率-失真优化可以提高压缩性能。

修复 ME		无/有 MVC		我们的	
Bpp	PSNR	Bpp	PSNR	Bpp	PSNR
0.044	27.33	0.20	24.32	0.029	28.17

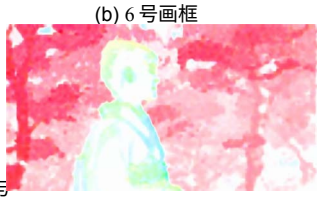
表 1：光流编码的比特成本和不同设置下光流扭曲帧的相关 PSNR。

(a) 5 号框架

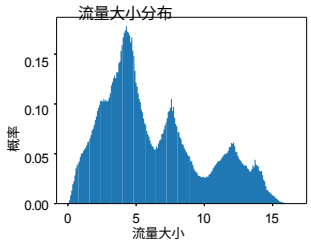
(b) 6 号画框

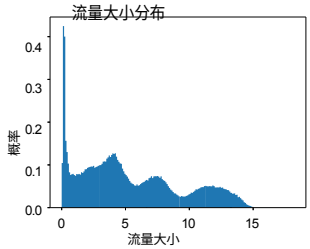


(c) 重建后的光流 (d) 重建后的光流与 fixing ME Net.



联合培训战略。





(e) 运算放大器的振幅分布 (f) 运算放大器的振幅分布
自然流向图 (c)。 声流图（d）。

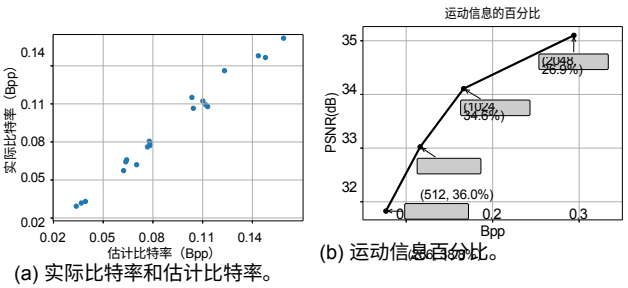
在图 7 中，我们进行了进一步的直观比较。图 7 (a)和(b)分别代表猕猴桃果实中的第 5 帧和第 6 帧。

图 7：流程可视化和统计分析。

单声道序列。图 7 (c) 表示在训练过程中固定光流网络时重建的光流图。图 7 (d) 表示使用联合训练策略后重建的光流图。图 7 (e) 和 (f) 是相应的光流幅度概率分布。可以看出，使用我们的方法重新构建的光流图包含了更多的光流幅度为零的像素（例如在人体区域）。虽然在这些区域，零值并不是真正的光流值，但我们的方法仍能在同质区域生成精确的运动补偿结果。更重要的是，有更多零值光流图所需的编码比特要少得多。，对图 7（c）中的光流图进行编码需要 0.045bpp，而对图 7（d）中的光流图进行编码只需要 0.038bpp。

值得一提的是，在 H.264 [39] 或 H.265 [31]中，为了达到更好的压缩性能，很多运动矢量都被赋值为零。令人惊讶的是，我们的预设框架可以学习类似的运动分布，而无需像 [39, 31] 那样依赖任何复杂的手工制作的运动估计策略。

运动补偿。在本文中，运动补偿系统将对运动进行补偿。



(a) 实际比特率和估计比特率。 (b) 运动信息百分比。

图 8：比特率分析。

在估计光流的基础上，利用运动补偿网络完善扭曲帧。为了评估该模块的效果，我们在另一项实验中取消了拟议系统中的运动补偿网络。以 *W/O MC* 为基础的替代方法的实验结果（见图 6 中的绿色曲线）显示，在相同的 bpp 水平下，没有运动补偿网络的 PSNR 将下降约 1.0 dB 。

更新策略。如第 3.6 节所述，在对当前帧 x_t 进行编码时，我们会在训练阶段使用一个在线缓冲区来存储之前重建的帧 $\hat{x}_{t-1}$。我们还报告了在训练阶段直接将上一重建帧 $\hat{x}_{t-1}$ 与上一原始帧 x_{t-1} 重新放置时的压缩性能。图 6 显示了 *W/O 更新*替代方法的结果（见红色曲线）。这表明，在相同的 bpp 水平下，缓冲策略可以提供约 0.2dB 的增益。

MV 编码器和解码器网络。在我们提出的框架中，我们设计了一个 CNN 模型来压缩光流并编码相应的运动表示。在不使用任何 CNN 的情况下，直接量化原始光流值并对其进行编码也是可行的。我们取消了 MV 编码器和解码器网络，进行新的实验。图 6 中的实验结果表明，去掉运动压缩网络后，以 *W/O MVC* 表示的替代方法（见洋红色曲线）的 PSNR 将下降 2 dB 以上。此外，表 1 还提供了在这种情况下进行光流编码的比特成本以及相应的翘曲帧 PSNR（以 W/O MVC 表示）。很明显，直接对原始光流值进行编码需要更多的比特（0.20Bpp），相应的 PSNR（24.43dB）也比我们提出的方法（28.17dB）差很多。因此，在使用光流估算运动时，对运动进行压缩至关重要。

运动信息。在图 2(b)中，我们还研究了只保留剩余编码器和解码器网络的设置。与我们的方法相比，在不使

我们提出的端到端视频通信框架的参数数量约为 11M。为了测试不同编解码器的速度，我们使用配备英特尔至强 E5-2640 v4 CPU 和单 Ti- tan 1080Ti GPU 的计算机进行了实验。对于分辨率为 352x288 的视频，Wu *等人*的工作[40]中每次迭代的编码（*或*解码）速度为 29fps（*或* 38fps），而我们的总体速度为 24.5fps（*或* 41fps）。H.264 和 H.265 的相应编码速度基于 offi-offi-offi-offi。

用任何运动估计方法的情况下独立处理每个帧（见黄色曲线，表示 *W/O 运动信息*）会导致 PSNR 下降超过 2dB 。

运行时间和模型复杂性运行时间和模型复杂度

商业软件 JM [2] 和 HM [3] 的编码速度分别为 2.4fps 和 0.35fps。商业软件 x264 [6] 和 x265 [7] 的编码速度分别为 250fps 和 42fps。虽然商业编解码器 x264 [6] 和 x265 [7] 的编码速度比我们的快得多，但它们需要大量的代码优化。相应地，最近的深度模型压缩方法是现成的，可以使深度模型的速度更快，这超出了本文的范围。

比特率分析。本文使用 [12] 中的概率定时网络来估算编码运动信息和残差信息的比特率。为了验证其可靠性，我们在图 8(a) 中使用算术编码比较了估计比特率和实际比特率。很明显，估计码率与实际码率接近。此外，我们还进一步研究了比特率的组成部分。在图 8(b)中，我们提供了每个点的λ 值和运动信息的百分比。当目标函数 $\lambda\ D*+$变大时，整个 Bpp 也会变大，而相应的运动信息百分比则会下降。

5. 结论

在中，我们提出了用于视频压缩的端到端深度学习框架。我们的框架既继承了传统视频压缩标准中经典预测编码方案的优点，又具有 DNN 强大的非线性表示能力。实验结果表明，我们的方法优于广泛使用的 H.264 视频压缩标准和最新的基于学习的视频压缩系统。这项工作为将深度神经网络应用于视频压缩提供了一个前景广阔的框架。基于所提出的框架，其他用于光流、图像压缩、双向预测和速率控制的新技术也可以很容易地插入到这个框架中。

致谢 本研究得到了国家自然科学基金（61771306）、上海市自然科学基金（18ZR1418100）、国家科技重大专项（2013ZX01033001-002-002）、上海市数字媒体处理与传输重点实验室（STCSM 18DZ2270700）的部分资助。

参考资料

[1] F. Bellard，bpg 图像格式。http://bellard.org/bpg/。访问时间：2018-10-30。1, 2

[2] h.264/avc 参考软件。http://iphome.hhi.de/suehring/。访问时间：2018-10-30.8

[3] Hevc 测试模型 (hm)。https://hevc.hhi.fraunhofer.de/HM-doc/。访问时间：2018-10-30.8

[4] 超视频组测试序列。http://ultravideo。cs.tut.fi.访问时间：2018-10-30。6

[5] Webp. https://developers.google.com/speed/webp/.访问时间：2018-10-30。3

[6] x264, 最佳 最佳 编码器。 编码器。 https://www.videolan.org/developers/x264.html. 访问时间：2018-10-30。8

[7] x265 hevc 编码器 / h.265 视频编解码器。http://x265.org。Accessed: 2018-10-30.8

[8] E.Agustsson, F. Mentzer, M. Tschannen, L. Cavigelli、R.Timofte, L. Benini, and L. V. Gool.端到端学习可压缩代表的软硬向量量化。In *NIPS*, pages 1141-1151, 2017.1, 2, 5

[9] E.Agustsson, M. Tschannen, F. Mentzer, R. Timo- fte, and L. Van Gool.用于极端学习图像压缩的生成对抗网络》，*arXiv 预印本 arXiv:1804.02958*, 2018.1, 2

[10] M.H. Baig、V. Koltun 和 L. Torresani。学习内画用于图像压缩。In *NIPS*, pages 1246-1255, 2017.2

[11] J.Balle´, V. Laparra, and E. P. Simoncelli. 端到端优化图像压缩。*arXiv preprint arXiv:1611.01704*, 2016.1, 2, 4, 5

[12] J.Balle´, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston.使用尺度超优先级的变分图像压缩。*arXiv preprint arXiv:1802.01436*, 2018.1, 2, 5, 8

[13] T.Chen, H. Liu, Q. Shen, T. Yue, X. Cao, and Z. Ma.Deep-coder：基于深度神经网络的视频压缩。In *VCIP*, pages 1-4.IEEE, 2017.2

[14] Z.Chen, T. He, X. Jin, and F. Wu.Learning for video compression. *arXiv preprint arXiv:1804.09869*, 2018.2

[15] A.Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas、V.Golkov 、 P. Van Der Smagt 、 D. Cremers 和 T. Brox.Flownet：用卷积网络学习光流。In *ICCV*, pages 2758-2766, 2015.3

[16] S.Han、H. Mao 和 W. J. Dally。 Deep compres-sion：用剪枝、训练量化和胡夫曼编码压缩深度神经网络》，*arXiv preprint arXiv:1510.00149*, 2015.1

[17] T.-W. Hui, X. Tang, and C. Change Loy.Hui, X. Tang, and C. Change Loy.Liteflownet：用于光流检测的轻量级卷积神经网络。In *Proceedings of the IEEE Conference on Com-puter Vision and Pattern Recognition*, pages 8981-8989, 2018.3

[18] T.-W. Hui, X. Tang, and C. C. Loy.A lightweight optical flow cnn-revisiting data fidelity and regularization.轻量级光流 cnn-重访数据保真度和正则化。*arXiv preprint arXiv:1903.07414*, 2019.3

[19] N.Johnston, D. Vincent, D. Minnen, M. Covell, S. Singh、T.Chinen、S. Jin Hwang、J. Shor 和 G. Toderici。针对递归网络的改进型有损图像压缩（带priming和空间自适应比特率）。In *CVPR*, June 2018.1, 2

[20] D.P. Kingma 和 J. Ba.Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.6

[21] M.Li, W. Zuo, S. Gu, D. Zhao, and D. Zhang.用于内容加权图像压缩的计算网络学习（Learning con- volutional networks for content-weighted image compres- sion）。In *CVPR*, June 2018.1, 2

[22] Z.Liu, X. Yu, Y. Gao, S. Chen, X. Ji, and D. Wang.利用卷积神经网络实现 hevc 硬线内编码器的 Cu par- tition 模式决策。*tip*, 25(11):5088-5103, 2016.
2

[23] G. Lu、W. Ouyang、D. Xu、X. Zhang、Z. Gao 和 M.-T. Sun。Sun.用于减少视频压缩伪影的深度卡尔曼滤波网络。In *ECCV*, September 2018.2

[24] F.Mentzer, E. Agustsson, M. Tschannen, R. Timofte, and L.Van Gool.用于深度图像压缩的条件概率模型。In *CVPR*, number 2, page 3, 2018.2

[25] D.Minnen, G. Toderici, M. Covell, T. Chinen, N. Johnston、J.Shor, S. J. Hwang, D. Vincent, and S. Singh.使用平铺深度网络的空间自适应图像压缩。In *ICIP*, pages 2796-2800.IEEE, 2017.2

[26] C.V. 网络指数。2016-2021年预测与方法》白皮书。*美国加利福尼亚州圣何塞*，2016 年 1 月。1

[27] A.Ranjan 和 M. J. Black.使用空间金字塔网络进行光流估计。In *CVPR*, volume 2, page 2.IEEE, 2017.3, 6

[28] O.Rippel 和 L. Bourdev.实时自适应图像通信。In *ICML*, 2017.1, 2

[29] A.Skodras, C. Christopoulos, and T. Ebrahimi.jpeg 2000 静态图像压缩标准》。*IEEE Signal Pro- cessing Magazine*，18（5）：36-58，2001.1, 2

[30] R.Song, D. Liu, H. Li, and F. Wu.基于神经网络的 hevc 内预测模式算术编码。In *VCIP*, pages 1-4.IEEE, 2017.2

[31] G. J. Sullivan, J.-R.Ohm、W.-J. Han、T. Wiegand 等. 高效视频编码（hevc）标准概述.
TCSVT, 22(12):1649-1668, 2012.1, 2, 3, 4, 5, 6, 7

[32] D.Sun, X. Yang, M.-Y. Liu, and J. Kautz.Liu, and J. Kautz.Pwc-net：使用金字塔、翘曲和成本量的光流 Cnns。In *CVPR*, pages 8934-8943, 2018.3

[33] L.Theis, W. Shi, A. Cunningham, and F. Husza´r.使用压缩自动编码器的有损图像压缩》，*arXiv 预印本 arXiv:1703.00395*, 2017.1, 2

[34] G. Toderici, S. M. O'Malley, S. J. Hwang, D. Vincent、D.Minnen, S. Baluja, M. Covell, and R. Sukthankar.使用递归神经网络的可变速率图像压缩》，*arXiv preprint arXiv:1511.06085*, 2015.1, 2, 5

[35] G. Toderici、D. Vincent、N. Johnston、S. J. Hwang、D. Min-nen、J. Shor 和 M. Covell。利用递归神经网络进行全分辨率图像压缩。In *CVPR*, pages 5435- 5443, 2017.1, 2

[36] Y.-H. Tsai, M.-Y.Tsai, M.-Y. Liu, D. Sun, M.-H.Liu, D. Sun, M.-H. Yang, and J. Kautz.Yang, and J. Kautz.为特定领域视频流学习二进制残差表示。*第三十二届 AAAI 人工智能大会*，2018 年。2

[37] G. K. Wallace.jpeg 静态图片压缩标准》。*IEEE Transactions on Consumer Electronics*，38（1）：xviii-xxxiv，1992.1, 2

[38] Z.用于图像质量评估的多尺度结构相似性。In *ASILOMAR CONFERENCE ON SIGNALS SYSTEMS AND COMPUTERS*, volume 2, pages 1398-1402.IEEE; 1998, 2003.2, 6

[39] T.Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra.H. 264/AVC 视频编码标准概述。*TCSVT*，13（7）：560-576，2003.1, 2, 3, 4, 5, 6, 7

[40] C.-Y. Wu, N. Singhal, and P. Krahenbuhl.Wu, N. Singhal, and P. Krahenbuhl.通过图像插值实现视频压缩。In *ECCV*, September 2018.3, 6, 8

[41] C.-Y. Wu, M. Zaheer, Hu, R. Manmatha, A. J. Smola, and P. Kra¨henbu¨hl.Wu, M. Zaheer, H. Hu, R. Manmatha, A. J. Smola, and P. Kra¨henbu¨hl.压缩视频动作识别。In *CVPR*, pages 6026-6035, 2018.1

[42] T.Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman.面向任务流的视频增强》，*arXiv preprint arXiv:1711.09078*, 2017.2, 5