

Implicit Neural Representations for Image Compression 论文笔记

1. 论文基本信息

Implicit Neural Representations for Image Compression

Yannick Strümpfer^{*1}, Janis Postels^{*1}, Ren Yang¹, Luc Van Gool¹, and Federico Tombari^{2,3}

¹ ETH Zurich
² Technical University of Munich
³ Google

发布于: *ECCV 2022*

2. 创新点

- 提出基于INR的压缩元学习初始化，加速了压缩速率和能产生更好的率失真性能。
- 将 SIREN 网络与位置编码相结合，提高了率失真性能。
- 为了强调基于 INR 的源压缩的通用性，我们进一步对 **3D 形状压缩**进行了实验，其中我们的方法大大优于传统的压缩算法 Draco。

3. 背景

论文中采用源压缩中有损压缩的概念（有损压缩旨在权衡文件的质量及其大小——称为率失真权衡），因为 **无损压缩** 的理论极限是香农定理中提到的熵。

RDAEs是一种特殊类型的自动编码器（Autoencoder）。在机器学习中，自动编码器是一种**神经网络** 结构，用于学习数据的紧凑表示（编码），然后尽可能地重构原始数据（解码）。Rate-Distortion Autoencoders（率失真自动编码器）的目标是联合优化两个方面：编码的速率（Rate）和重构的失真（Distortion）。文章中的INRs摒弃了这种方式，隐式神经表示是一种灵活的多用途的数据表示，能够适用于不同分变率的数据。

论文的结论：

论文引入了一个压缩管道，它大大优于最近提出的 **COIN**，并且与传统的图像压缩算法具有竞争力。此外，论文中证明了元学习INRs在下采样图像上已经优于JPEG2000和一些RDAE。最后，通过将论文中的方法直接应用于 3D 数据压缩来强调基于 INR 的**图像压缩** 的普遍性，其中论文中的算法优于传统的算法 Draco。

难点：

(1) 即使使用最简单的传统算法，直接方法也很难竞争。

(2) 由于INRs通过对特定实例过拟合对数据进行编码，因此编码时间被认为是不切实际的。为此，论文做出了两个贡献。首先，我们提出了基于 INR 的压缩元学习。我们利用基于模型不可知元学习(MAML)的INRs元学习的最新进展来寻找权重初始化，可以用更少的梯度更新压缩数据，并产生更好的率失真性能。其次，我们将SIREN与位置编码相结合。

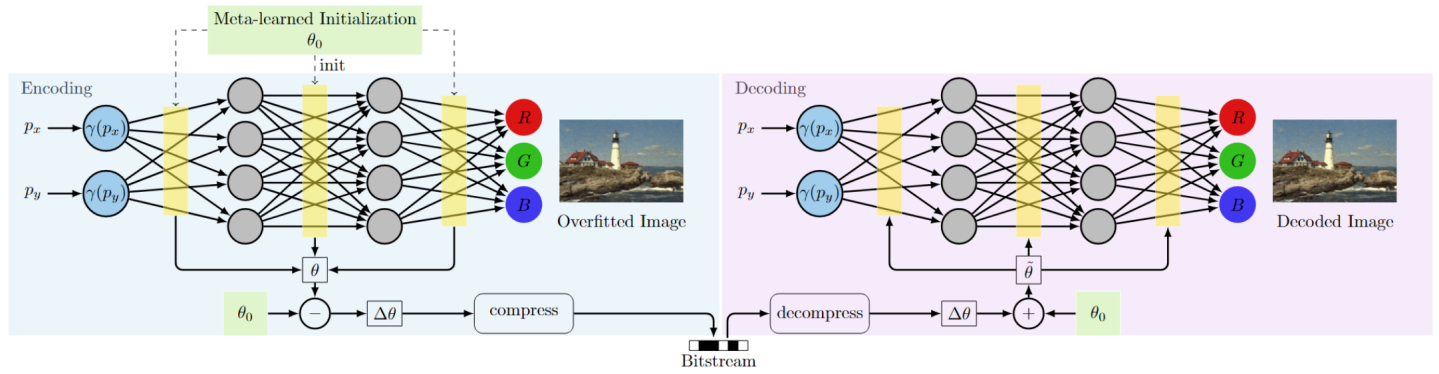
名词解释：

- 元学习：元学习的目标是让**模型学会如何学习**。具体来说，元学习让模型在学习过程中变得更智能，能够适应新的、以前未见过的任务。好比是在训练一个学习算法，而不仅仅是一个完成特定任务的模型。
- 元学习的目标是让**模型学会如何学习**。具体来说，元学习让模型在学习过程中变得更智能，能够适应新的、以前未见过的任务。好比是在训练一个学习算法，而不仅仅是一个完成特定任务的模型。
- 率失真性能：
 - 率 (Rate)**：表示压缩的效率，通常以比特率（bits per second）的形式表示。更高的比特率意味着更高的压缩速度，但也可能伴随着更高的数据量。
 - 失真 (Distortion)**：表示压缩后与原始数据之间的差异，即信息的损失。更低的失真表示压缩后的数据质量更接近原始数据，但可能需要更多的比特来表示。
 - 在保持合理的压缩率的同时尽量减小失真。提高率失真性能通常表示压缩算法在相同的压缩率下能够保持更高的数据质量，或者在相同的数据质量下能够实现更高的压缩率。

INRs:

INRS是通过将数据表示为从坐标到值的连续函数来存储基于坐标的数据，例如图像、视频和3D形状。

4. Pipeline



4.1. 基于隐式神经表示的压缩

$$\arg \min_{\theta} \mathcal{L}(\mathbf{x}, f_{\theta}(\mathbf{p})) = \theta^* \xrightarrow{\text{transmit } \theta^*} \hat{\mathbf{x}} = f_{\theta^*}(\mathbf{p}).$$

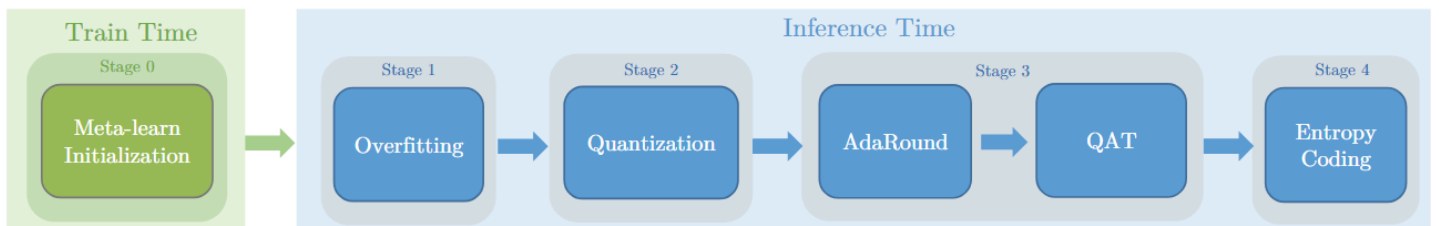
INRs 在网络权重 θ 中隐式存储所有信息。INR 本身的输入是坐标 \mathbf{p} ，不包含任何信息。编码过程相当于训练 INR。解码过程相当于将一组权重加载到网络中并在坐标网格上进行评估。传输过程中只需要存储 θ^* ，网络架构上采用 SIREN，使用 $\omega = 30$ 的正弦激活的 MLP。

而这个输入的 \mathbf{p} 通过位置编码在这个编码中，引入了一个尺度参数 σ 以调整频率间隔，并将频率项与原始坐标 \mathbf{p} 进行连接（就像在 SIREN 代码库中一样）。该编码的形式可以表示为：

$$\gamma(\mathbf{p}) = (\mathbf{p}, \sin(\sigma^0 \pi \mathbf{p}), \cos(\sigma^0 \pi \mathbf{p}), \dots, \sin(\sigma^{L-1} \pi \mathbf{p}), \cos(\sigma^{L-1} \pi \mathbf{p})).$$

其中 L 是使用的频率数量。这个编码的目的是将原始的一维坐标 \mathbf{p} 转换为包含不同频率的高维向量 $\gamma(\mathbf{p})$ ，以便在模型中更好地捕捉输入的复杂结构。

4.2. 使用随机初始化的基本方法



4.2.1. 第1阶段：过拟合

第一阶段，输入单张图片使得 INR 网络学习该图片的特征，过拟合学习该单张图片，给定一个图像 \mathbf{x} 和一个坐标网格 \mathbf{p} ，最小化目标：

$$\arg \min_{\theta} \mathcal{L}_{\text{MSE}}(\mathbf{x}, f_{\theta}(\mathbf{p})).$$

。使用均方误差来衡量真实目标和 INRs 输出的相似性

$$\mathcal{L}_{\text{MSE}}(\mathbf{x}, \hat{\mathbf{x}}) = \sum_i^W \sum_j^H \frac{\|\mathbf{x}_{ij} - \hat{\mathbf{x}}_{ij}\|_2^2}{WH}.$$

4.2.2. 第2阶段：量化

由于过拟合而产生的模型权重是单精度浮点数，每个权重需要32位。为了减少内存需求，作者使用了 AI Model Efficiency Toolkit (AIMET) 对权重进行了量化。7-8 范围内的位宽导致我们模型的最佳率失真性能。

4.2.3. 第3阶段：量化后优化

AdaRound 优化：使用 AdaRound 方法，这是一种二阶优化方法，用于智能地决定是将权重向上舍入还是向下舍入。这样做的目的是更精细地控制量化误差，以提高性能。

Quantization Aware Training (QAT)：在量化后，通过 QAT 对权重进行微调。这个步骤的目标是在一定程度上纠正量化误差，提高模型的性能。由于量化是不可微的，作者使用 Straight Through Estimator (STE) 来计算梯度，以便在反向传播期间绕过量化操作。

4.2.4. 第4阶段：熵编码

使用二值化算数编码来无损压缩量化的权重。

熵编码中目前较为成熟的有霍夫曼编码和算术编码，以及14年Duda提出的ANS（非对称系数）编码。

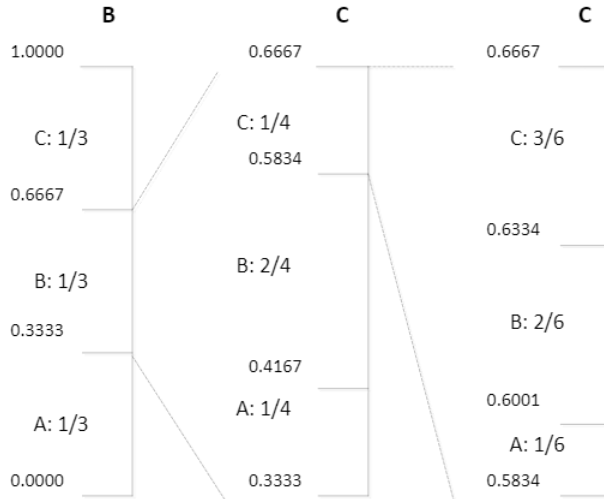
算术编码是一种无限接近熵编码理论值的编码，其本质操作就是用一个 $[0, 1)$ 的小数来表示最终的编码结果，其基本操作也是基于统计来进行的。例如：

当前编码的字符是ABC，如何编码“BCC”这个字符？

答：1) 设定初始频率值，三种字符概率均值分布，则均为 $1/3 \rightarrow 0.333$ 。

2) 输入B，其位于 $[0.333, 0.667)$ ，则以此区间进行下一次划分（由于输入了B，原来的三个字符变成四个 A B B C），概率变成 A 0.25 B 0.5 C 0.25，以此继续划分。

3) 最终得到编码的区间，输出该区间内的小数，转换为二进制数即为最终的编码结果。



4.3. 元学习初始化压缩INRs

通过元学习（小规模任务上的快速训练，以得到一个初始的模型参数）来初始化一组参数 θ_0 ，学习到的参数在权重空间中更加接近最终的INR，因此假设更新 $\Delta\theta = \theta - \theta_0$ 比更新全权重 θ 需要更少的存储，因此我们可以固定 θ_0 包含它在解码器中，只传输 $\Delta\theta$ 。解码端：

$$\tilde{\theta} = \theta_0 + \Delta\tilde{\theta}, \quad \hat{\mathbf{x}} = f_{\tilde{\theta}}(\mathbf{p}).$$

为了在训练过程中只更新 $\Delta\theta$ 在MLP中的所有线性层应用分解（实质上分解成两个线性层固定了一层）：

$$\begin{aligned} \mathbf{W}\mathbf{x} + \mathbf{b} &= (\mathbf{W}_0 + \Delta\mathbf{W})\mathbf{x} + (\mathbf{b}_0 + \Delta\mathbf{b}) \\ &= \underbrace{(\mathbf{W}_0\mathbf{x} + \mathbf{b}_0)}_{\text{fix}} + \underbrace{(\Delta\mathbf{W}\mathbf{x} + \Delta\mathbf{b})}_{\text{quantize \& retrain}}. \end{aligned}$$

5. 💎 实验成果展示

5.1. 评价指标

1. **比特率**：测量为存储表示所需的比特数除以图像的像素个数 $W \cdot H$ ：

$$\text{bitrate} = \frac{\text{total number of bits}}{WH} \quad [\text{bpp}].$$

1. **PSNR**：据 MSE 测量失真，并使用以下公式将其转换为峰值信噪比 (PSNR)：

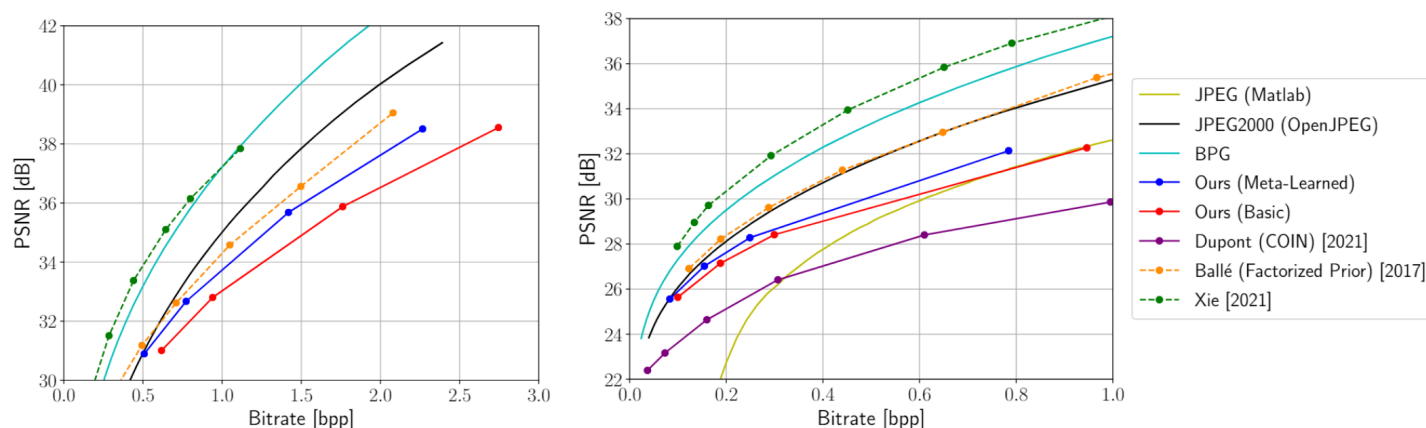
$$\text{PSNR} = 10 \log_{10} \left(\frac{1}{MSE} \right) \quad [\text{dB}].$$

5.2. Baselines

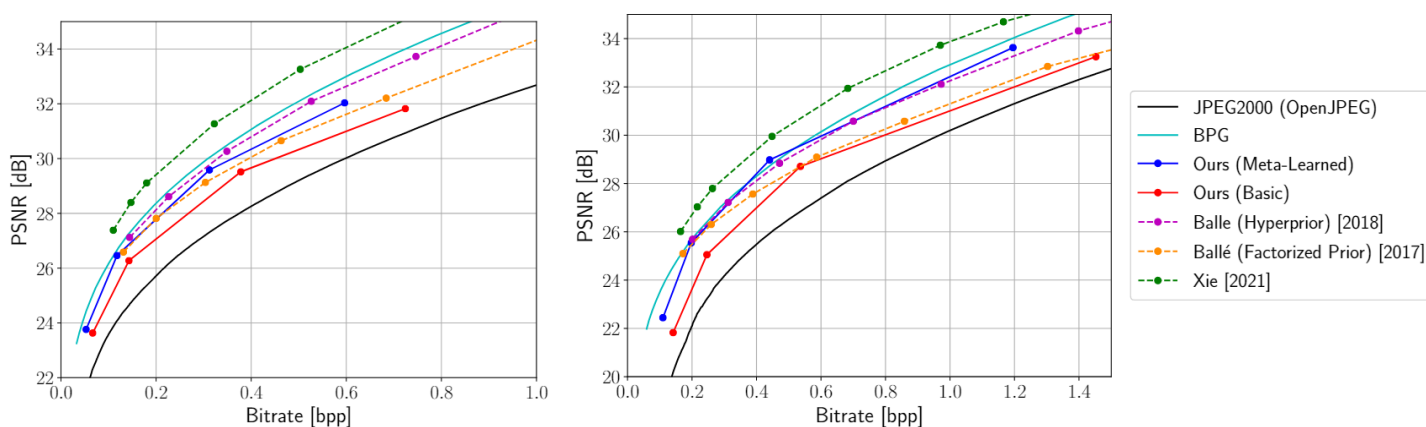
Baselines. We compare our method against traditional codecs, INR based compression and learned approaches based on RDAEs.

- Traditional image compression codecs: JPEG, JPEG2000, BPG
- INR-based image compression: Dupont *et al.* [19] (COIN)
- RDAE-based image compression: Ballé *et al.* [6], Xie *et al.* [58]
- 3D mesh compression: Draco ⁴
















5.3. 结果



在CelebA(左)柯达(右)数据集上评估的传统(实线)、学习的自动编码器(虚线)和学习的INR方法(带点实线)的图像压缩方法的性能。



图像压缩方法，包括传统的(实线)、RDAE(虚线)和学习的基于INR的方法(带点实线)，在柯达数据集上进行评估，图像分辨率降低了两倍(左)和四个(右)。元学习的INRs在这种情况下表现出具有竞争力的性能。

original	Draco (7 bit)	Draco (6 bit)	Ours ($M = 128$)	Ours ($M = 64$)
 407.5 MB	 2.2 MB	 1.9 MB	 57.0 KB	 16.9 KB
 3.4 MB	 54.8 KB	 45.9 KB	 57.4 KB	 17.2 KB
 32.8 MB	 283.0 KB	 231.7 KB	 54.8 KB	 16.6 KB

与应用于 3D 形状压缩的方法相比，网格压缩算法 [Draco](#) 的视觉比较。使用论文的方法比较了 Draco 的 2 个量化设置，即 6 位和 7 位，以及两个隐藏维度 $M = 64$ 、 128 。我们的方法在相似或更低的存储下显示出明显更平滑的表面重建和更好的细节。