Modality-Agnostic Variational Compression of Implicit Neural Representations

Schwarz[1,2], Tack[3], Teh[1], Lee[4], Shin[3]

[1] Deepmind, [2] University College London, [3] KAIST, [4] POSTECH

ML Reading Group, 08.02.2023

Presentation Structure

- Why strive for modality-agnostic compression algorithms?
- A refresher on implicit neural representations
- Using implicit neural representations for compression
- Improving implicit neural representations for compression
    - Improved conditioning
    - Improved compression
- Experiments
    - Effectiveness of improved conditioning
    - Effectiveness of improved compression
- Conclusion

Why strive for modality-agnostic compression algorithms?

- Currently, custom compression techniques for each modality
  - E.g. MP3 for audio, JPEG for images, HEVC for video, and so on
- Each carefully introduce inductive biases that help in the respective modality
- This limits the transfer of algorithmic ideas between these techniques
- Sometimes, scientists need to collect data for which no generally accepted compression technique may even be available

Paradigm shift:

Make modality agnosticism a **key guiding principle**

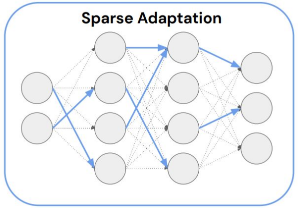
→ research advancements are now more widely applicable

A refresher on implicit neural representations

- Interpret data as functions from coordinates to features
  - E.g. (x,y) → (r,g,b) for images
- Parameterize these functions with neural networks, e.g. functa [Functa]
- INRs are inherently modality agnostic
  - Always applicable if data can be expressed as a coordinate to feature mapping
  - Obviously the case for image, voxels, scene, climate, audio and video datasets
  - Side note: extensions also exist for e.g. graphs [GeneralizedINRs]
- In the end, a data point is encoded within the weights of a neural network
  - How to store those weights efficiently?
    → Previous works propose e.g. quantizing the weights
  - *Data*-compression becomes *model*-compression

Using implicit neural representations for compression

- Reminders:
  - An INR is a function $f(\cdot;\, \theta):\ C \rightarrow Y$
  - Can be optimized using the mean-squared error: $\min_{\boldsymbol{\theta}} \sum_{j=1}^{M} \|f(\mathbf{c}_j; \boldsymbol{\theta}) - \mathbf{y}_j\|_2^2.$
  - Bad idea to do separately per datapoint
  - We instead use data-item specific parameters ϕ^i that are used to specialize a shared INR $f(\cdot; \boldsymbol{\theta})$ that captures structure across the dataset. θ Is typically much larger than the ϕ^i
- How to condition/specialize the f on the ϕ^i?
  - Commonly: layer-wise modulations, i.e. $\phi^i = \left[s^{(1)}, \ldots, s^{(L)}\right]$
  - $\mathbf{c}^{(l-1)} \mapsto h(\mathbf{W}^{(l)}\mathbf{c}^{(l-1)} + \mathbf{b}^{(l)} + \mathbf{s}^{(l)})$
- Reducing the size of ϕ^i further: Two common options
  - Predict the $s = \left[s^{(1)}, \ldots, s^{(L)}\right]$ from ϕ^i using shared weights, i.e. $\mathbf{s} = \mathbf{W}'\phi + \mathbf{b}'$ [Functa]
    - But: hard to train, so far limited to linear mappings → lack of representational capacity
  - Prune dimensions in ϕ^i through sparsity [MSCN]
    - But: requires approximate inference, introduces additional complexity&hyperparameters



Improving implicit neural representations for compression

- INR-based compression can be improved with a two-fold approach:
  (i) Improved conditioning : Try to achieve high signal reconstruction pre-quantization
  (ii) Improved compression: Better quantization techniques
- These are orthogonal algorithmic considerations
  - (i) increases the upper-bound of performance we can hope to achieve after quantisation
  - (ii) reduces the gap between that upper-bound and the actual final performance
- Improved conditioning:
  - Recent approaches use either sparsity or latent coding for small representations
  - They propose a middle ground
    - Learns more efficiently, and also
    - Provides better reconstructions at equal capacity
- Improved compression:
  - Introduce a *learned* compressor
  - It is trained on the latent representations of the training dataset items
  - Can then be applied on unseen latent representation to compress them

Improved conditioning: INR specialisation through subnetwork selection

- Combine both ideas of sparsity and parametric predictions
  - Sparsity, but without hard gating. This alleviates the need for approximate inference
  - Parametric predictions can concentrate capacity on non-sparse entries of s
- Propose a non-linear prediction network that maps a ϕ^i to one $\mathbf{G}_{\texttt{low}}^{(l)}$ per layer
- $\mathbf{G}_{\texttt{low}}^{(l)}$ is a low-rank soft gating mask, same shape as weights of layer l
- One layer now applies the following function:

SIREN

elementwise mult.

$$\mathbf{c}^{(l-1)} \mapsto \sin(\omega_0(\mathbf{G}_{\texttt{low}}^{(l)} \odot \mathbf{W}^{(l)}\,\mathbf{c}^{(l-1)} + \mathbf{b}^{(l)}))$$

activations of previous layer

$$\mathbf{G}_{\texttt{low}}^{(l)} := \sigma(\mathbf{U}^{(l)}\mathbf{V}^{(l)\top}),$$

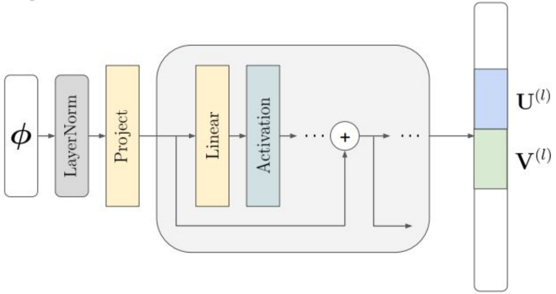
sigmoid

Local (from ϕ^i)
Global (from θ)

Improved conditioning:
INR specialisation through subnetwork selection

- $$\mathbf{c}^{(l-1)} \mapsto \sin(\omega_0(\mathbf{G}^{(l)}_{\texttt{low}} \odot \mathbf{W}^{(l)}\ \mathbf{c}^{(l-1)} + \mathbf{b}^{(l)}))$$
$$\mathbf{G}^{(l)}_{\texttt{low}} := \sigma(\mathbf{U}^{(l)}\mathbf{V}^{(l)\top}), \quad \boxed{\mathbf{U}^{(l)}, \mathbf{V}^{(l)} \in \mathbb{R}^{m \times d} \quad \text{with } d \ll m}$$

- Similar to previous techniques, $\mathbf{U}^{(l)}, \mathbf{V}^{(l)}$ are not directly the entries of ϕ^i
- Instead, they are the output of a deep residual network with input ϕ^i
  → Its parameters are part of θ



$$\mathbf{G}^{(l)}_{\texttt{low}} := \sigma\left(\mathbf{U}^{(l)}\ \mathbf{V}^{(l)T}\ \ddots\right)$$

(a) Non-linear projection from ϕ to $\mathbf{G}^{(l)}_{\texttt{low}}$ sub-network gates.

Improved conditioning:
INR specialisation through subnetwork selection

- The whole thing is optimized using model-agnostic Meta Learning [MAML]
- How to compress a new (unseen) test datapoint? $= \sum_{j=1}^{M} \| f(c_j; \theta; \phi_0) - y_j \|_2^2$
  → Compute its representation as $\phi = \phi_0 - \alpha \nabla_{\phi_0} \mathcal{L}_{\text{INR}}(\boldsymbol{\theta}, \phi_0, \mathbf{x})$ (inner loop)
- Key idea of MAML: backpropagate through this optimisation process
  → Thereby we learn an initialization ϕ_0 and the global parameters $\boldsymbol{\theta}$
- We thus optimize $\min_{\boldsymbol{\theta}, \phi_0} \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})} \left[\mathcal{L}_{\text{INR}}(\boldsymbol{\theta}, \phi_0 - \alpha \nabla_{\phi_0} \mathcal{L}_{\text{INR}}(\boldsymbol{\theta}, \phi_0, \mathbf{x}), \mathbf{x}) \right]$ (outer loop)
- Additionally, they also meta-learn the step-size α, as in Meta-SGD[MetaSGD]
- Drawbacks: requires a lot of memory (due to 2nd order gradients)
  - Thus they need to use patches for large data. E.g. they only compress 32x32 blocks of pixels
  - There are 1st order methods, but previous work found they severely hinder performance

Improved compression: Variational compression of modulations

- They adapt the method in "End-to-end optimized image compression" [End2End]
  - That method was devised for image data, does not make use of INRs
  - Learns to encode images as a code with low rate and good reconstructions after quantization
  - Basically, it's a variational autoencoder under a specific generative and inference model
- But.. we want model-agnosticity!
  - The authors apply this same method, not to learn to compress inputs, but the modulations ϕ^i
  - The results are quantised discrete codes that can be stored by e.g. Huffmann coding
- The optimized compression loss is a weighted sum of rate and distortion

analysis transform / "encoder" quantized code

  - $$\begin{aligned}\mathcal{L}_{\texttt{compress}}(\boldsymbol{\pi}_a, \boldsymbol{\pi}_s, \mathbf{x}, \phi) &= \mathcal{L}_{\texttt{rate}} + \lambda \mathcal{L}_{\texttt{distortion}} \\ &= -\log_2[p_{\hat{\mathbf{z}}}(Q(g_a(\phi; \boldsymbol{\pi}_a)))] + \lambda \mathcal{L}_{\texttt{MSE}}(g_s(\hat{\mathbf{z}}; \boldsymbol{\pi}_s), \phi)\end{aligned}$$

entropy model quantizer $\hat{z}$ = round(z) synthesis transform / "decoder"

  - Note that we can set $\mathcal{L}_{\texttt{distortion}} = \mathcal{L}_{\texttt{MSE}}(f(\cdot; \boldsymbol{\theta}, g_s(\hat{\mathbf{z}}; \boldsymbol{\pi}_s)), \mathbf{y})$

Experimental results
Effectiveness of improved conditioning

Experiments: Effectiveness of advanced conditioning

Dataset	Model	Performance @ dim(ϕ) 64	128	256	512	1024
ERA5 (4×)	Functa	43.2	43.7	43.8	44.0	44.1
	MSCN	44.6	45.7	46.0	46.6	46.9
	VC-INR	**45.0**	**46.2**	**47.6**	**49.0**	**50.0**
CelebA-HQ	Functa	21.6	23.5	25.6	28.0	30.7
	MSCN	21.8	23.8	25.7	28.1	**30.9**
	VC-INR	**22.0**	**23.9**	**26.0**	**28.3**	30.8
SRN Cars	Functa	22.4	23.0	23.1	23.2	23.1
	MSCN	22.8	**24.0**	**24.3**	24.5	24.8
	VC-INR	**23.9**	**24.0**	**24.3**	**25.2**	**25.5**
ShapeNet10	Functa	$_{99}.30$	$_{99}.40$	$_{99}.44$	$_{99}.50$	$_{99}.55$
	MSCN	$_{99}.43$	$_{99}.50$	$_{99}.56$	$_{99}.63$	$_{99}.69$
	VC-INR	$_{99}$**.54**	$_{99}$**.61**	$_{99}$**.64**	$_{99}$**.70**	$_{99}$**.71**

Metric: PSNR: $-10 \cdot \log_{10}(MSE)$

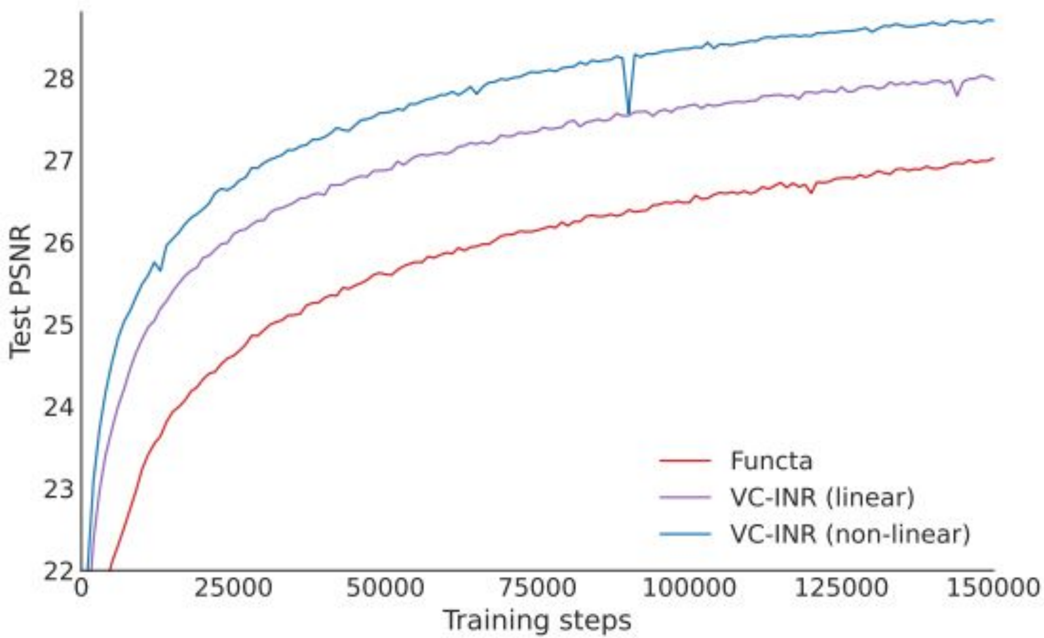
Metric: Voxel Accuracy

PSNR	MSE
50	1.00E-05
40	1.00E-04
30	1.00E-03
20	1.00E-02

Model: 15 layers of 512 neurons each

Experiments: Effectiveness of advanced conditioning

- Is the usage of a non-linear mapping from ϕ^i to the modulations useful?



(a) Learning curves

Experiments: Effectiveness of advanced conditioning

- Does the mask $\mathbf{G}_{\texttt{low}}$ condition the shared INR on image statistics?

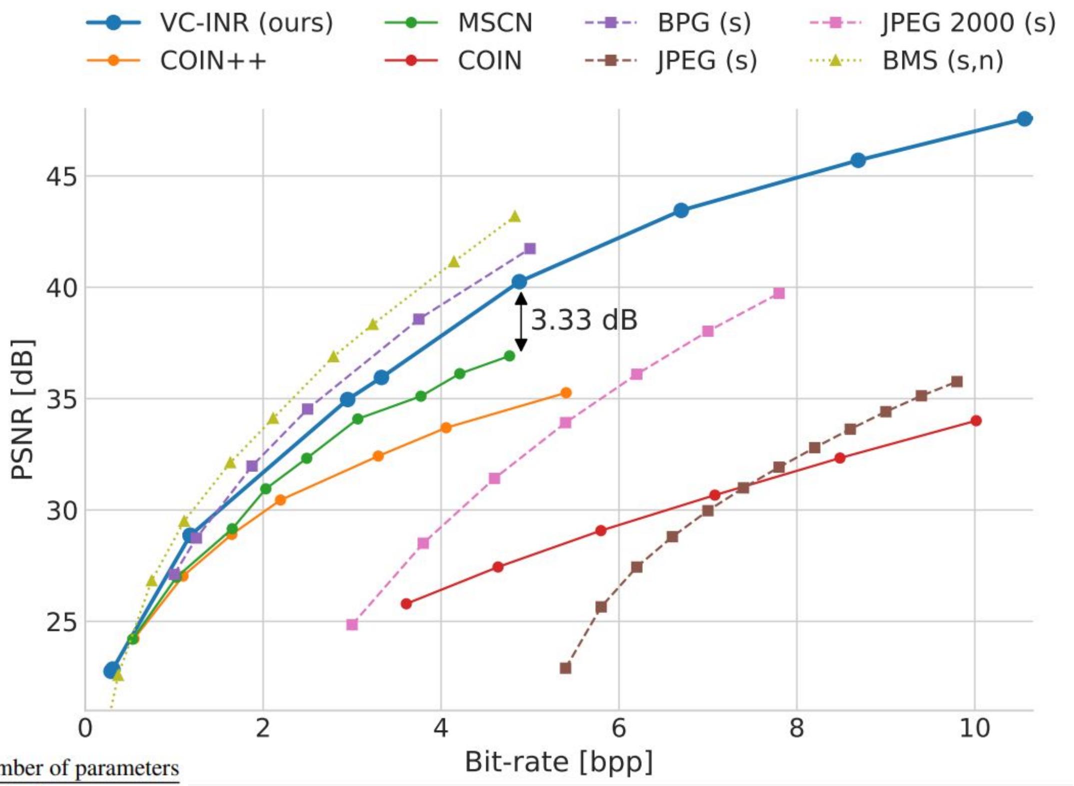
(c) Mask clustering on CelebA-HQ

Experimental results
Effectiveness of improved compression

Experiments: Data compression across modalities: Images

(CIFAR-10)

Dashed line ≙ modality-specific, (n) ≙ neural compression

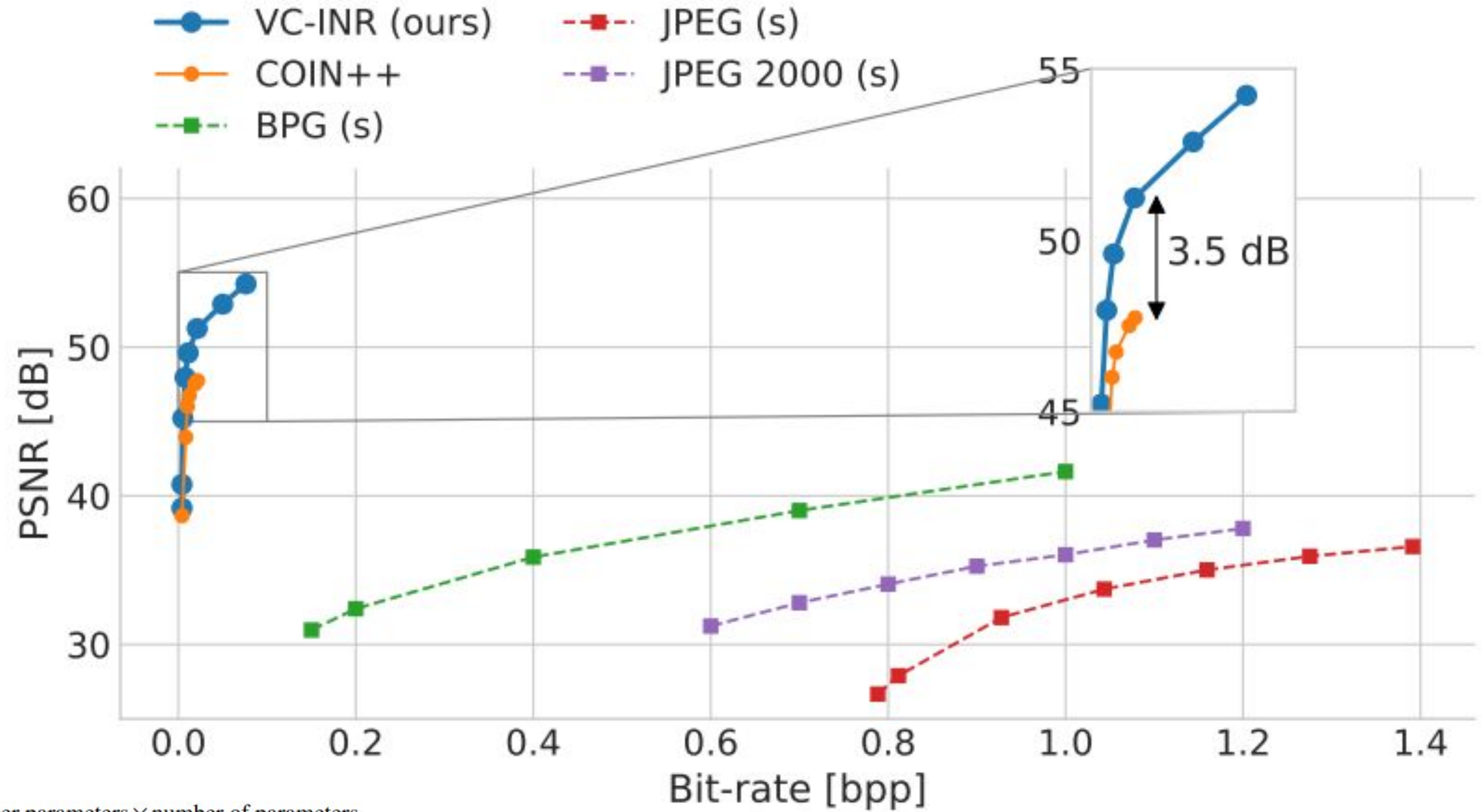


$$\text{bpp} = \frac{\text{bits per parameters} \times \text{number of parameters}}{\text{number of pixels}}$$

Experiments: Data compression across modalities: Manifold

(ERA5)

Dashed line ≙ modality-specific

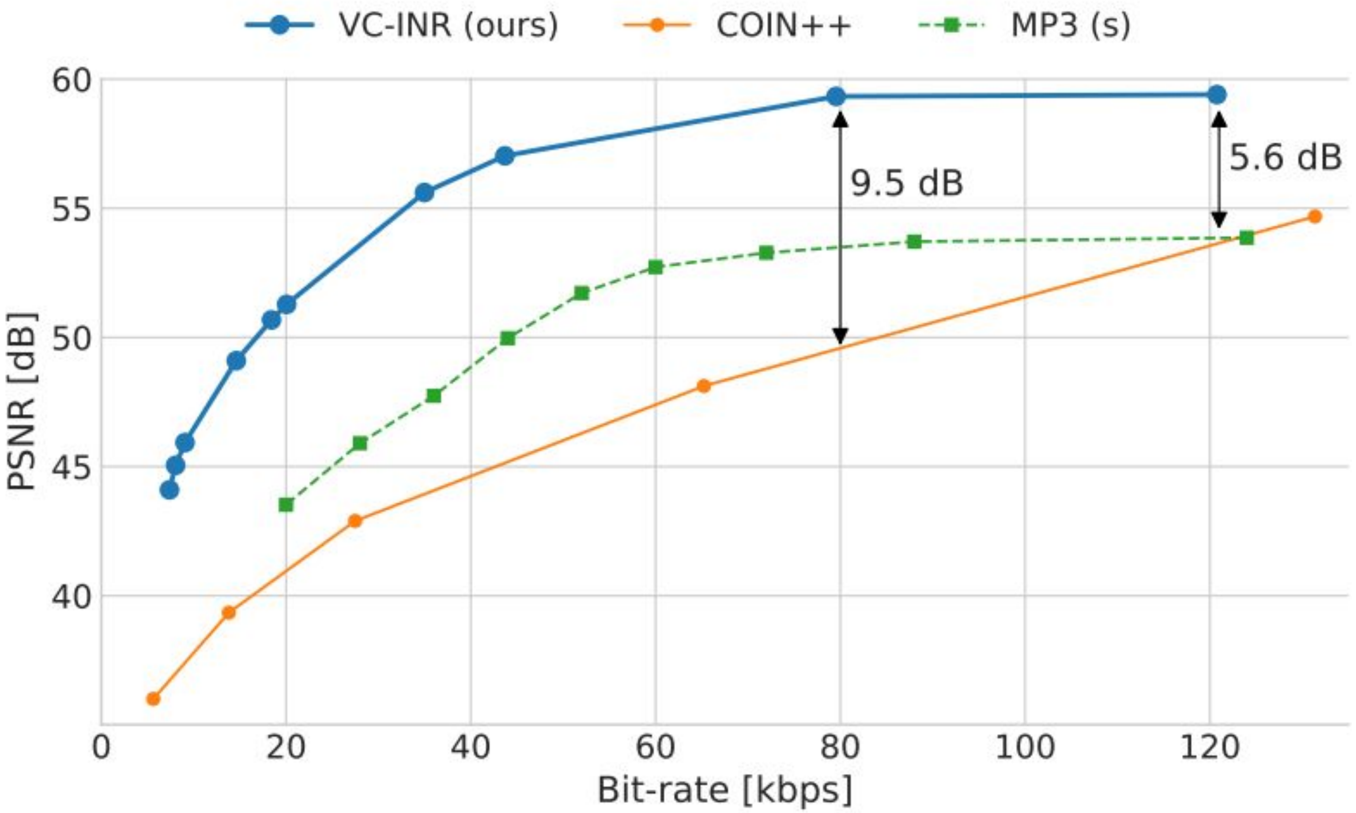


$$\text{bpp}=\frac{\text{bits per parameters}\times\text{number of parameters}}{\text{number of pixels}}$$

Experiments: Data compression across modalities: Audio

(LibriSpeech)

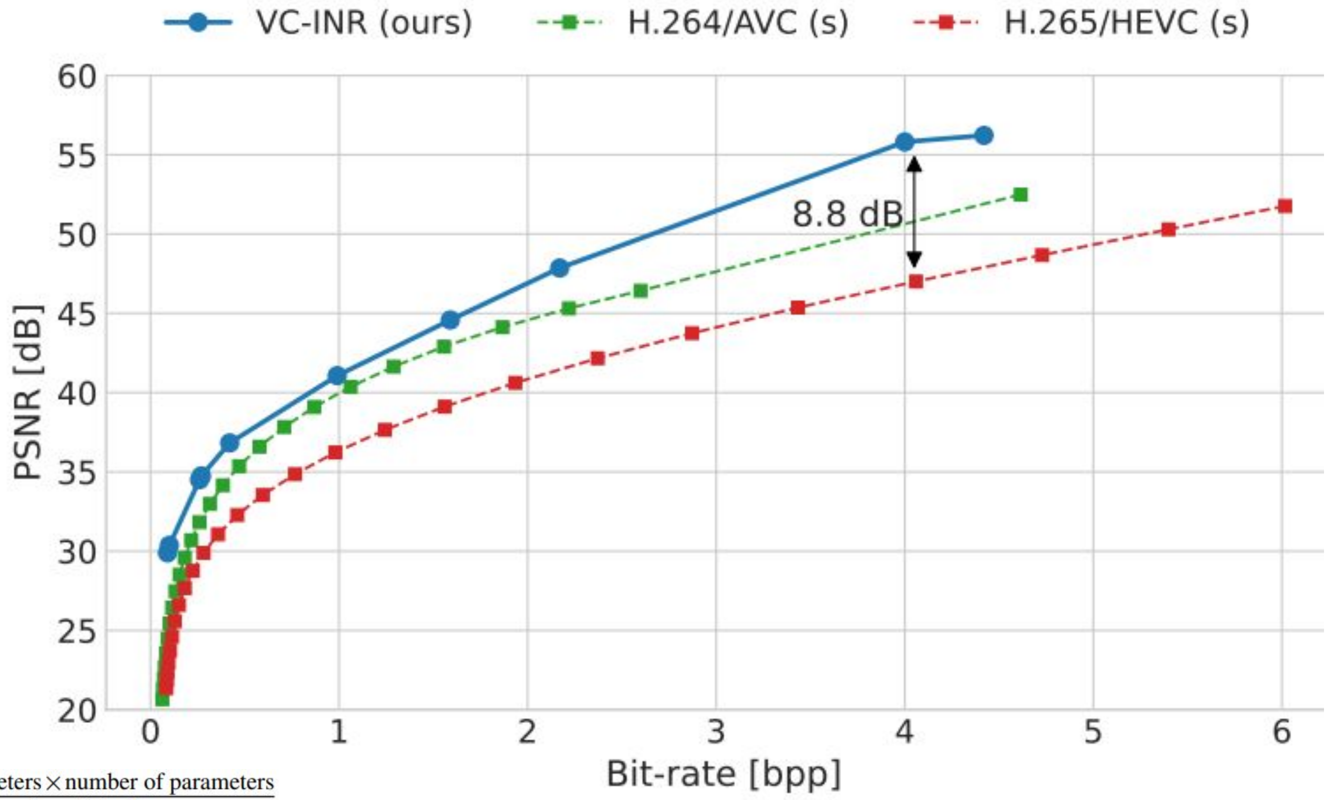
Dashed line ≙ modality-specific



Experiments: Data compression across modalities: Video

(UCF-101)

Dashed line ≙ modality-specific



$$\text{bpp} = \frac{\text{bits per parameters} \times \text{number of parameters}}{\text{number of pixels}}$$

Conclusion

- The authors introduce VC-INR, a modality-agnostic neural compression technique
- They make modality-agnosticity a key guiding principle
- They propose algorithmic improvements across both conditioning and compression
- For improving conditioning, they combine ideas from latent modulation and sparsity
- For improving compression, they apply a previous neural compression method to the modulations
- They sometimes even outperform modality-specific codecs such as JPEG and MP3

Thank you for listening :-)

References I

- [VC-INR] Modality-Agnostic Variational Compression of Implicit Neural Representations
- [Functa] From data to functa: Your data point is a function and you can treat it like one
- [GeneralizedINRs] Generalised Implicit Neural Representations
- [MSCN] Meta-Learning Sparse Compression Networks
- [SIREN] Implicit Neural Representations with Periodic Activation Functions
- [MAML] Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks
- [MetaSGD] Meta-sgd: Learning to learn quickly for few-shot learning
- [Coin++] COIN++: Neural Compression Across Modalities
- [End2End] End-to-end optimized image compression

Algorithm 1 INR Meta-training stage

Data: Dataset $\{\mathbf{x}^i, \mathbf{y}^i\}_{i=1}^N$

Initialise shared network $\boldsymbol{\theta}$ and latent modulation initialisation ϕ_0.

while *not converged* **do**

  Sample batch of data $\mathcal{B} = \{\mathbf{x}^j, \mathbf{y}^j\}_{j=1}^B$

  `// Adaptation loop (`$\mathcal{O}$` in Figure 1c)`

  for $j \leftarrow 1$ **to** B **do**

    `// For 1 adaptation step`

    $\phi^j \leftarrow \phi_0 - \alpha \nabla_{\phi_0} \mathcal{L}_{\texttt{MSE}}(f(\mathbf{x}^j, \boldsymbol{\theta}, \phi_0), \mathbf{y}^j)$

  `// Update using adapted latent modulation`

  $\phi_0 \leftarrow \phi_0 - \beta \mathbb{E}[\nabla_{\phi_0} \mathcal{L}_{\texttt{MSE}}(f(\mathbf{x}^j, \boldsymbol{\theta}, \phi^j), \mathbf{y}^j)]$

  `// Remaining INR parameters`

  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \beta \mathbb{E}[\nabla_{\boldsymbol{\theta}} \mathcal{L}_{\texttt{MSE}}(f(\mathbf{x}^j, \boldsymbol{\theta}, \phi^j), \mathbf{y}^j)]$

Result: Dataset of latent modulations $\{\phi^i\}_{i=1}^N$, $\boldsymbol{\theta}$

Algorithm 2 Quantisation training stage

Data: Dataset of latent modulations $\{\phi^i\}_{i=1}^N$, $\boldsymbol{\theta}$, λ

while *not converged* **do**

  Initialise parameters $\boldsymbol{\pi}_a, \boldsymbol{\pi}_s$.

  Sample batch of data $\mathcal{B} = \{\boldsymbol{\phi}^j, \mathbf{x}^j, \mathbf{y}^j\}_{j=1}^B$

  for $j \leftarrow 1$ **to** B **do**

    $\mathbf{z} \leftarrow g_a(\boldsymbol{\phi}^j; \boldsymbol{\pi}_a)$

    `// Rounding at inference to obtain` $\hat{\mathbf{z}}^j$

    $\widetilde{\mathbf{z}}^j = \mathbf{z}^j + \boldsymbol{\epsilon}; \boldsymbol{\epsilon} \sim \mathcal{U}(-\frac{1}{2}, \frac{1}{2})$

    `// Compute entropy model` $p_{\hat{\mathbf{z}}}$ `and rate`

    $\ell^j_{\texttt{rate}} = -\log_2[p_{\hat{\mathbf{z}}}(\widetilde{\mathbf{z}})]$

    $\widetilde{\boldsymbol{\phi}}^j \leftarrow g_s(\widetilde{\mathbf{z}}^j; \boldsymbol{\pi}_s)$

    $\ell^j_{\texttt{distortion}} = \mathcal{L}_{\text{MSE}}(f(\mathbf{x}^j, \boldsymbol{\theta}, \widetilde{\boldsymbol{\phi}}^j), \mathbf{y}^j)$

  $\boldsymbol{\pi}_a \leftarrow \boldsymbol{\pi}_a - \beta\mathbb{E}[\nabla_{\boldsymbol{\pi}_a}(\ell^j_{\texttt{rate}} + \lambda\ell^j_{\texttt{distortion}})]$

  $\boldsymbol{\pi}_s \leftarrow \boldsymbol{\pi}_s - \beta\mathbb{E}[\nabla_{\boldsymbol{\pi}_s}(\ell^j_{\texttt{rate}} + \lambda\ell^j_{\texttt{distortion}})]$

COIN++

- Modulations: $\sin(\omega_0(W\mathbf{h} + \mathbf{b} + \boldsymbol{\beta}))$
  - Uses latent modulation with a **linear** transform
- Also meta-learns with MAML
- No compression of the latent modulation vector
- For quantisation, simply uses a uniform quantisation
- Then also applies entropy coding to store losslessly