

D’OH: Decoder-Only Random Hypernetworks for Implicit Neural Representations

Cameron Gordon¹, Lachlan E. MacDonald², Hemanth Saratchandran¹, and
Simon Lucey¹

¹ Australian Institute for Machine Learning,
University of Adelaide, Adelaide SA 5000, Australia
`{first.last}@adelaide.edu.au`

² Mathematical Institute for Data Science,
John Hopkins University, Baltimore MD 21218, USA

Abstract. Deep implicit functions have been found to be an effective tool for efficiently encoding all manner of natural signals. Their attractiveness stems from their ability to compactly represent signals with little to no offline training data. Instead, they leverage the implicit bias of deep networks to decouple hidden redundancies within the signal. In this paper, we explore the hypothesis that additional compression can be achieved by leveraging redundancies that exist *between* layers. We propose to use a novel runtime decoder-only hypernetwork – that uses no offline training data – to better exploit cross-layer parameter redundancy. Previous applications of hypernetworks with deep implicit functions have employed feed-forward encoder/decoder frameworks that rely on large offline datasets that do not generalize beyond the signals they were trained on. We instead present a strategy for the optimization of runtime deep implicit functions for single-instance signals through a Decoder-Only randomly projected Hypernetwork (D’OH). By directly changing the latent code dimension, we provide a natural way to vary the memory footprint of neural representations without the costly need for neural architecture search on a space of alternative low-rate structures.

Keywords: Implicit Neural Representations · Compression · Hypernetworks

1 Introduction

Implicit neural representations (INRs), also known as coordinate networks, have received attention for their ability to represent signals from different domains – including sound, images, video, signed distance fields, and neural radiance fields – within a signal-agnostic framework [15, 22, 55, 56, 74, 87]. When combined with quantization strategies, they act as a neural signal compressor which can be applied generally across different modalities – the best example of which is Compressed Implicit Neural Representations (COIN) [22]. Combining this with lossless entropy compression can lead to even further reductions [18, 22, 34, 68, 75].

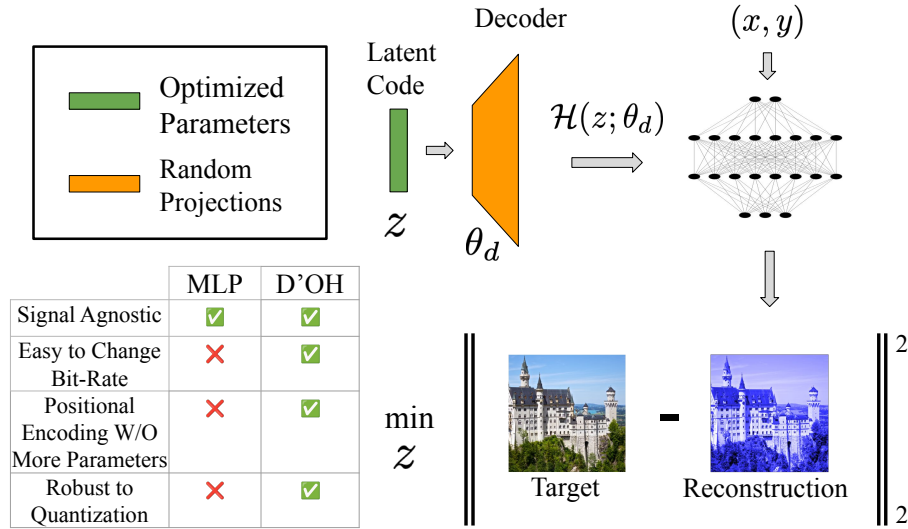


Fig. 1: Illustration of the proposed Decoder-Only Hypernetwork, which optimizes a low-dimensional latent code z to generate weights for the target implicit neural representation (INR). This signal-agnostic framework operates without offline training data, relying solely on the target architecture and the specific data instance. Random projections act as the network decoder θ_d , facilitating a compact code representation.

Neural networks are believed to contain significant parameter redundancy, motivating interest in the use of more compact architectures [21, 33, 35, 39]. Hypernetworks (meta-networks that generate the weights of a target network) are a popular approach to approximate the performance of more expressive architectures [27, 35]. Often they are pre-trained on a target signal class or application – for example, super-resolution, in-painting, or style transfer [5, 14, 43, 57]. These hypernetworks are offline **encoders** of the characteristics of the signal class, which are runtime **decoded** to the target network weights (see: Figure 2). Often they require more complex architectures than the target network - using highly-parameterized transformers or graph neural networks to predict simple MLPs [14]. By using external data these hypernetworks can generate new networks for a target class of signals. However, this comes at the cost of compactness, the need for offline optimization, and a reduced ability to generalise to out-of-distribution data instances – primary motivations for the use of INRs [22, 47, 61].

Hypernetwork research in INRs has largely focused on pre-trained **encoder-decoder** hypernetworks, which use data instances as an input to produce a new INR. In contrast, we suggest that a **decoder-only** form of hypernetwork is possible in which the data instance appears only as the target output. These Decoder-Only Hypernetworks (D’OH) can be optimised directly to predict a given data instance by projecting to a target implicit neural network structure (see: Figure 1). Similar to standard INRs this requires no offline training on the

target signal class and can instead be optimised at runtime. We provide a simple model involving a low-dimensional parameter vector projected by a fixed linear random mapping to construct a target SIREN network [74]. By controlling a latent dimension for the hypernetwork and transmitting the random seed used to reconstruct the mapping to transmit the signal, we can use fewer parameters than the target network architecture – enabling its use in compression. In contrast to existing INR compression methods such as COIN [22, 75], which requires search on low-rate architectures or aggressive quantization to reduce bit-rate, our method keeps the same target architecture and instead directly varies latent dimension to compress to different rates. Furthermore, as SIRENs are highly sensitive to initialization [63], and standard initializations must be adapted for hypernetworks [13], we develop a novel initialization scheme for this approach.

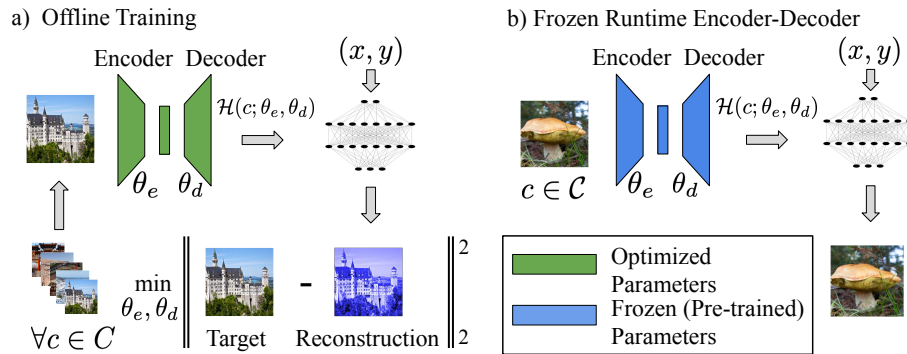


Fig. 2: Conventional Encoder-Decoder Hypernetworks are optimized offline on a signal class. The hypernetwork $[\theta_e, \theta_d]$ is frozen and runtime predicts INR weights for new data instances, limiting generality for out-of-distribution signals. In contrast, a Decoder-Only Hypernetwork (see: Figure 1) is runtime optimized using only the target instance.

We make the following contributions in this paper:

1. We introduce a signal-agnostic framework (**D’OH - Decoder-Only random Hypernetworks**) for the runtime optimization of implicit neural representations which requires optimization only on a target signal instance, with no need for offline training on an example set from the target signal class.
2. We provide a simple Decoder-Only Hypernetwork architecture based on a trainable latent code vector and fixed random projection decoder weights. As only the latent code, biases, and an integer random seed need to be communicated to reconstruct signals we show this is applicable to data compression.
3. We derive a novel hypernetwork initialization to match the layer-weight variances of SIREN networks, which modifies the distribution of the non-trainable random weights rather than the optimizable parameters.

Finally, we present an intriguing result in Section 5 that using a random linear hypernetwork induces quantization smoothing, leading D’OH to achieve simi-

lar performance for both Post-Training Quantization (PTQ) and Quantization Aware Training (QAT). This result, although not a central contribution for the paper, is of practical interest due to the negligible training overhead of PTQ [31].

2 Background and Motivation

2.1 Implicit Neural Representation Compression

An implicit neural representation (INR) is a function $f_\psi(\mathbf{x}) \rightarrow \mathbf{y}$ mapping coordinates \mathbf{x} to features \mathbf{y} where ψ are parameter values, trained to closely approximate a target signal g such that $\|f_\psi(\mathbf{x}) - g(\mathbf{x})\| \leq \epsilon$. INRs are *signal-agnostic* and have been widely applied to represent signal types including images, sound, neural radiance fields, and sign distance fields [24, 51, 55, 56, 74, 75, 87]. Recently, INRs have garnered significant interest for their potential in implicit compression [22, 87]. A forward pass of a trained network produces a lossy reconstruction of an original signal instance [18, 22–24, 75]. Further compression can be achieved by exploiting the high-degree of redundancy known to exist in neural network weights [21, 35, 37, 53]. For example, by quantizing network weights [17, 18, 22, 24, 51, 73, 75], pruning [17, 46], inducing sparsity [67, 88], using neural architecture search, variational methods [68], hash-tables [77, 78], latent transformations [42, 45, 90] and applying entropy compression [24, 34, 42, 45, 75].

2.2 Neural Architecture Search

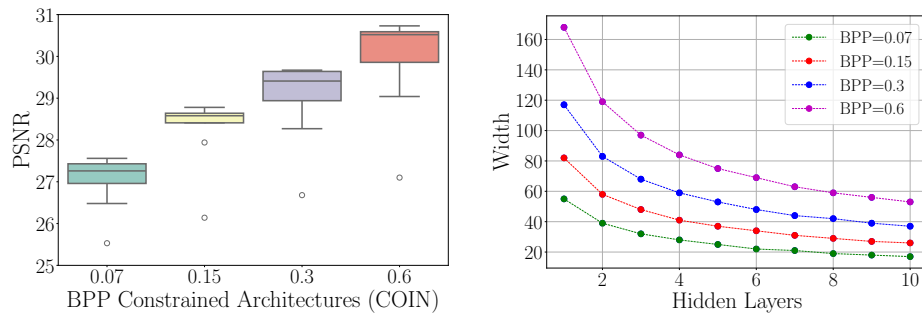


Fig. 3: Left: COIN architectures show variability in outcome necessitating a costly Neural Architecture Search to achieve maximal performance as in [22]. Right: Architectures that satisfy a bits-per-pixel constraint for COIN (Kodak index 2) using 16-bit quantization. There is a combinatorial increase in the search space for optimal architectures when considering different quantization levels (*e.g.* for quantization aware training (QAT) as in [18, 34, 75] as QAT requires fixed quantization during training [64]).

One of the difficulties in using INRs for compression is the need to decide between different architectures to achieve reductions in bit-rates [81]. In COIN

this is handled by performing Neural Architecture Search (NAS) [22, 25, 84]. MLP architectures (width, hidden) that satisfy a bits-per-pixel (BPP) target (e.g. 0.3) on the Kodak dataset [2] are searched over with the best performing models at each bit-rate selected for further experiments. Achieving similar bit-rates without a change in architecture would require aggressive quantization, which can severely degrade reconstruction accuracy. Figure 3 shows that NAS is useful for COIN, as architectures vary in performance at the same bit-rate. This search is costly, due to the time needed to train each model and the large number of satisfying architectures [81]. In contrast, by working with a fixed target architecture generated by a Decoder-Only Hypernetwork we can control the desired bit-rate by directly varying the latent code dimension. This reduces the search across alternative networks to a $O(1)$ decision for a target bit-rate.

2.3 Hypernetworks

A hypernetwork is a meta-network that generates the parameters of another network, typically referred to as the target network [14, 35]. As described by Ha *et al.* [35], this is achieved by approximating a target network’s architecture with a low-dimensional latent code and applying a generating function. For example:

$$\mathcal{H}_\theta(z) = \psi \tag{1}$$

where z is a latent code, θ are the hypernetwork weights, and ψ refers to the generated parameters of a target network function f_ψ . For simplicity, we will use W_l to refer to weights of the l^{th} layer of a L -layer multi-layer perceptron, and ψ to refer to parameters in a more generic mapping. Hypernetworks have been applied to various use-cases [14], including compression [28, 41, 57], neural architecture search [89], image super-resolution [43, 82] and sound generation [76]. There is no single dominant hypernetwork architecture and a wide number have been used in practice, including transformers; convolutional, recurrent, and residual networks; generative adversarial networks, graph networks, and kernel networks [14, 16, 28, 35, 70, 71, 89]. Hypernetworks have recently been applied to INRs, with the goal to generate task-conditioned networks suitable for use on new data instances [14]. Recent applications have included volume rendering [85], super-resolution of images [43, 85], hyperspectral images [91], sound [76], novel view synthesis [16, 69, 85], and partial-differential equations [20, 52].

2.4 Encoder-Decoder Hypernetworks

Hypernetworks as currently applied to INRs can be described as being *encoder-decoders*, where an example set drawn from the target signal class is used to condition a hypernetwork before being evaluated on a target signal instance. Given a class \mathcal{C} of a signal type, a training set $C \in \mathcal{C}$ with samples $c \in C$ is used to train a hypernetwork $\mathcal{H}(c; \theta_e, \theta_d)$ to generate the weights ψ of the implicit neural function $f_\psi(x)$ where x is the input coordinate. The network is optimized offline to minimise the loss $\sum_x \|f_\psi(x) - c(x)\|_2^2$ across the training set,

thereby learning characteristics of the signal class. At runtime the hypernetwork is used to generate the weights for an unseen data instance in the target class $c \in \mathcal{C}, c \notin \mathcal{C}$ for use a downstream task. We have used θ_e, θ_d to collectively define the parameters for encoding (learning from the data class) and decoding (generating the target network), but these can be part of a single hypernetwork architecture or more distinctly separated [14]. Examples of encoder-decoder hypernetworks for INRs include Klocek *et al.* [43] who train on DIV2K to produce INRs for image super-resolution [4]; Sitzmann *et al.* [74], who train on CelebA for image in-painting [29, 49]; Zhang *et al.* [91], who focus on hyperspectral images; and Szatkowski *et al.* [76], who train on VCTK to generate audio INRs. While not directly using a hypernetwork, several related works have sought to condition a base network on target class data through meta-learning techniques such as MAML [26], before using this as an initialization for fine-tuning, autoencoding, or learning a set of modulations for a novel data instance [24, 60, 68, 75, 79].

3 Methodology

3.1 Decoder-Only Hypernetworks

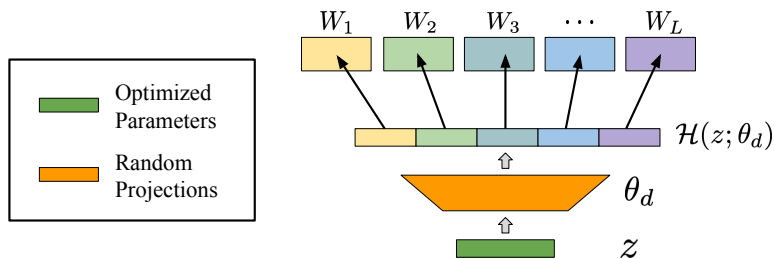


Fig. 4: Weight generation in a Decoder-Only Hypernetwork. Latent code z and decoding parameters θ_d are used to generate target network weights as the hypernetwork output $\mathcal{H}(z; \theta_d)$. We specifically investigate a random linear hypernetwork where $\mathcal{H}(z; \theta_d) = B_l z$ where B_l is a fixed and untrained per-layer random weight matrix.

While in Encoder-Decoder Hypernetworks the goal is to learn a domain-conditioned hypernetwork mapping to the target network, we instead propose to learn a latent code at runtime directly from the target data itself. As this process does not involve a pre-training stage of encoding domain information to the latent space, we describe this as a ‘Decoder-Only Hypernetwork’ in which the latent code is optimized runtime by directly projecting to a target network. A Decoder-Only Hypernetwork is denoted as $\mathcal{H}(z; \theta_d) \rightarrow \psi$. For our purposes we restrict ourselves to the situation in which the decoder parameters θ_d are fixed and only the latent code z is learned. As a specific example, we focus on a L -layer multi-layer perceptron target network and set the decoding weights to simply be

linear maps $\theta_d := \{B_l\}_{l=1}^L$ defined by a family of *fixed* random matrices B_l : one for each layer. The target network weights W_l for the l^{th} layer are defined to be the image of the latent vector $z \in \mathbb{R}^n$ under the linear map B_l . We maintain a separate trainable bias term h_l for each layer in the target network. Denoting \bar{W}_l as the generated vectorized form of W_l Equation (1) reduces to:

$$\mathcal{H}_\theta(z) = \{\bar{W}_l\}_{l=1}^L = \{B_l z\}_{l=1}^L. \quad (2)$$

This simple random projection architecture has a number of advantages. Firstly, it exploits depth-wise redundancy in the target network by inducing parameter sharing [14, 58, 66, 80], as the optimized parameters are tied by arbitrary sampled random matrices. Secondly, a random matrix decoder may be transmitted with a single random seed enabling a highly compact transmission protocol. A similar integer seed protocol was recently proposed by [44] in the context of Low-Rank Adaptation (LoRA). We note that more general decoder-only architectures could extend beyond the linear hypernetwork case we have proposed to include more expressive latent variables (*e.g.* per-layer latent variables z_l), different fixed mappings, or non-linear decoders. The auto-decoder approach of Park *et al.* [59] can be considered an example using a fully-parameterised decoder. The effective use of random matrices in the low-rank matrix decompositions described by Denil *et al.* [21], classical results applying random projections for dimension reduction [11, 19, 36, 40, 86], and recent works exploring compressibility of neural architectures using random projections motivate our approach [3, 6, 44, 54, 62].

3.2 Hypernetwork Training and Initialization

Initializing hypernetworks is non-trivial as standard schemes (such as He *et al.* [32] and Glorot *et al.* [38]) do not directly translate when parameters are contained in a secondary network [13]. As a result, without correction the target network may experience exploding or vanishing gradients [13], and important convergence properties are not guaranteed even under infinite-width hypernetworks [48]. Additionally, activations that are known to be sensitive to initialization, such as SIREN, require careful consideration even for standard MLPs [63, 74]. The common principle for initialization schemes is to set weights so that layer-output variance is preserved following activation [32, 38]. For hypernetworks we instead need to maintain the output variance for the target network. Chang *et al.* explore strategies for tanh and ReLU target networks by modifying parameter initialization in the hypernetwork. Linear hypernetworks as in Equation (2) using per-layer random projections and a shared latent code suggests a different strategy – changing the variance of the random projections. It can be shown³ that the variance of target network weights $\bar{W}_l = B_l z$ depends only on the variance of B_l , the variance of z , and the latent dimension n . We therefore set the random matrices to be uniformly initialized such that:

³ A full derivation and SIREN equivalent initialization is provided in Supplementary Materials 1.1

$$\text{Var}(B_l) = \frac{\text{Var}(\bar{W}_l)}{n\text{Var}(z)}. \quad (3)$$

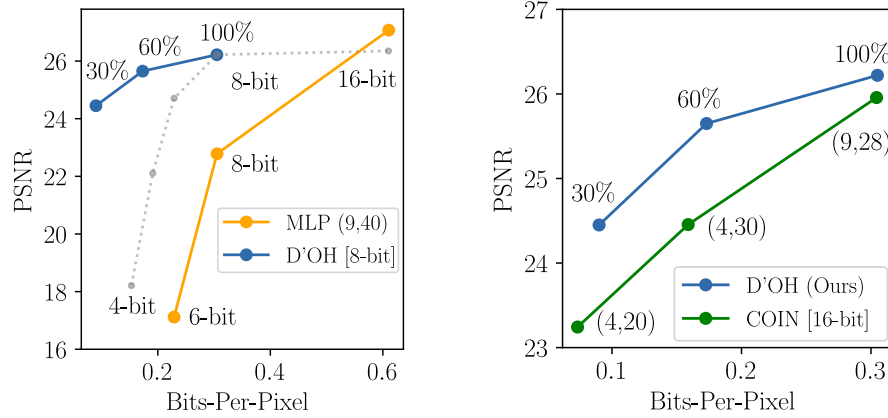
This ensures that the projected weight matrices will have identically distributed entries of the same variance as a given target network. In our experiments we initialize to preserve the variances of the original SIREN implementation [74]. As the SIREN input-layer is initialized separately to the rest of the network we would not be able to match all layers by initializing the latent code alone. We arbitrarily initialize the latent code uniformly between $[-1/n, 1/n]$. Biases are not tied to the latent code and are separately initialized to zero, excluding the output bias which is set to 0.5 in the middle of the output range.

3.3 Training Configurations and Metrics

We test the performance of D’OH on two implicit neural compression tasks: image regression and occupancy field representation. We choose a target network of 9 hidden layers and width 40, corresponding to the 0.6 BPP model for KODAK in [22]. We test latent code dimensions calculated to represent approximately 100%, 60%, and 30% of the parameters of the target (9, 40) MLP model as calculated without positional encoding. Note that the number of parameters increases for MLP models with positional encoding, but not with D’OH (see: Section 5 for discussion). For MLP baselines, we train models using the configurations in COIN at 0.07, 0.15, 0.3 BPP, corresponding to (width, hidden) pairs of (20,4), (30,4), (28, 9), and (40,9) [22]. See Supplementary Table 1 for the full set of configurations used for training. We experiment both with and without positional encoding (10 Fourier frequencies [56]). Following training we apply Post-Training Quantization to model weights at the best performing epoch and calculate the perceptual metrics at each quantization level. Compression metrics are reported using the estimated memory footprint (parameters \times bits-per-parameter) and the bits-per-pixel (memory/pixels). We find this to be a close proxy to the performance of an entropy compressor (BZIP2 [72]), with some variation due to file overhead. See Supplementary Materials 2 for technical details of the applied Post-Training Quantization strategy and compression.

4 Results

Image Compression - Kodak Image regression is a $\mathbb{R}^2 : \mathbb{R}^3$ coordinate function used to implicitly represent 2D images by predicting RGB values at sampled coordinates. We conduct image compression experiments on the Kodak dataset [2]. The Kodak dataset consists of natural images with dimensions of 768×512 pixels and is a common test for implicit neural image compression (e.g. [18, 22, 24, 75]). We test across the 24 image Kodak dataset and report rate-distortion performance for the 100%, 60%, and 30% latent code dimensions, using 8-bit PTQ. We select three literature Kodak benchmarks to highlight as examples of a signal specific codec (JPEG), a signal-agnostic and data-less codec



(a) D’OH is more robust to quantization (grey) than MLPs (orange). As such we can pick an optimal quantization level (8-bit) and directly varies the latent code to control bit-rate (blue).

(b) D’OH avoids Neural Architecture Search by maintaining the same target architecture, enabling a simple choice for a lower rate model.

Fig. 5: Reducing bit-rate for a fixed MLP requires either a) Aggressive Quantization, or b) Neural Architecture Search on a space of low-rate structures. Kodak dataset.

(COIN) [22], and a meta-learned signal-agnostic codec (COIN++) [24]; however, our most direct point of comparison is the original COIN as it relies on no external data or signal specific information. Figure 6 shows the performance of D’OH relative to comparison algorithms as reported by CompressAI [9]. D’OH shows improved rate-distortion performance relative to JPEG, COIN, and COIN++. In the Supplementary Materials 4.1 we provide additional benchmarks showing performance relative to leading compression methods that use domain information, including those incorporating meta-learned Quantization Aware Training [75], auto-encoders [7], and advanced image codecs [10].

Qualitative Qualitative results are shown in Figure 9. As COIN is fixed at 16-bits [22], we use an additional MLP benchmark to show direct comparison to D’OH at 8-bit quantization. We can note that D’OH greatly outperforms the 8-bit MLP models, potentially due to reduced quantization error when using a single latent code. This is consistent with Figure 5a. When compared to COIN (16-bit), which requires different architectures to vary bit-rate, D’OH outperforms at all comparable bit-rates while using the same target architecture. Note that in this experiment we use positional encoding for the D’OH model (which uses no additional parameters), and no positional encoding for the MLP models. In Supplementary Figure 3 we demonstrate reduced rate-distortion performance for very low-rate MLP image models with positional encoding, due to increased input layer parameters, so this is a stronger benchmark comparison.

Image Regression - DIV2K DIV2K is a 512×512 pixel natural image dataset frequently used for image regression [4]. We repeat the experiment for 10 indices

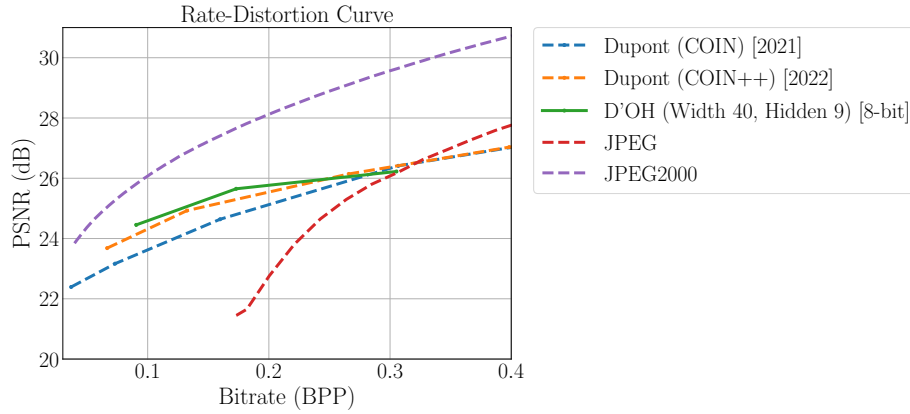


Fig. 6: Rate-Distortion curve on the Kodak dataset. D’OH approximates a single target architecture (Width 40, Hidden 9) with 8-bit Post-Training Quantization. The rate distortion curve is generated by varying the latent code dimension. The D’OH model exceeds the performance of JPEG, COIN, and COIN++.

of the DIV2K dataset, using the same D’OH and MLP configurations from the Kodak experiment, training for 500 epochs per instance, and applying 8-bit PTQ. Figure 7 shows the results evaluated for PSNR \uparrow , Shared-Structural Similarity (SSIM \uparrow) [83], and Learned Perceptual Image Patch Similarity loss (LPIPS \downarrow) [92]. Results show that D’OH performs better than the MLPs across each metric.

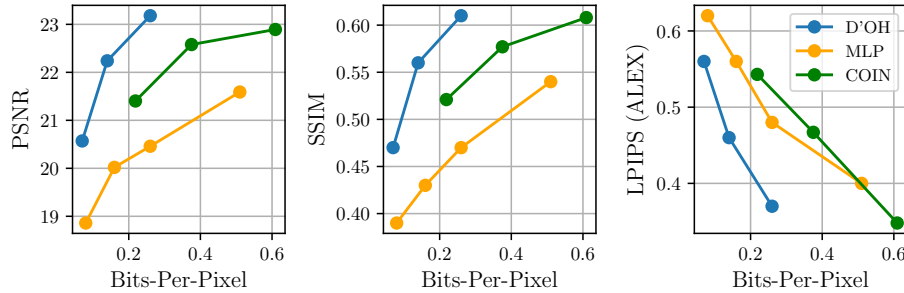


Fig. 7: DIV2K Image Regression with 8-bit PTQ. Results averaged over 10 indices. D’OH shows improved performance relative to the MLP. This corresponds to a BD-Rate of -78.66%, and BD-PSNR of 2.38db indicating substantial rate reduction, evaluated using the maximal performance at each rate and applying Akima interpolation [8, 12].

Occupancy Field Experiments A Binary Occupancy Field is a $\mathbb{R}^3 : \mathbb{R}^1$ coordinate function used to implicitly represent 3D shapes with the model output

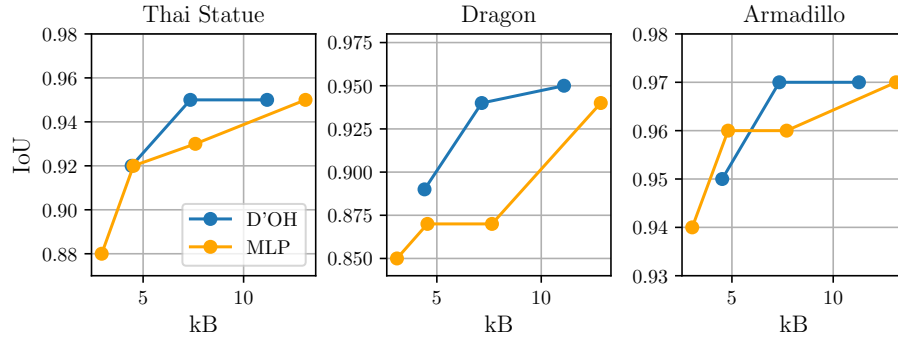


Fig. 8: Binary Occupancy Field experiments (6-bit quantization). D'OH exhibits improved rate-distortion improvement over quantized MLPs. Rate-distortion generated for D'OH by varying the latent dimension and by varying architecture for MLPs.

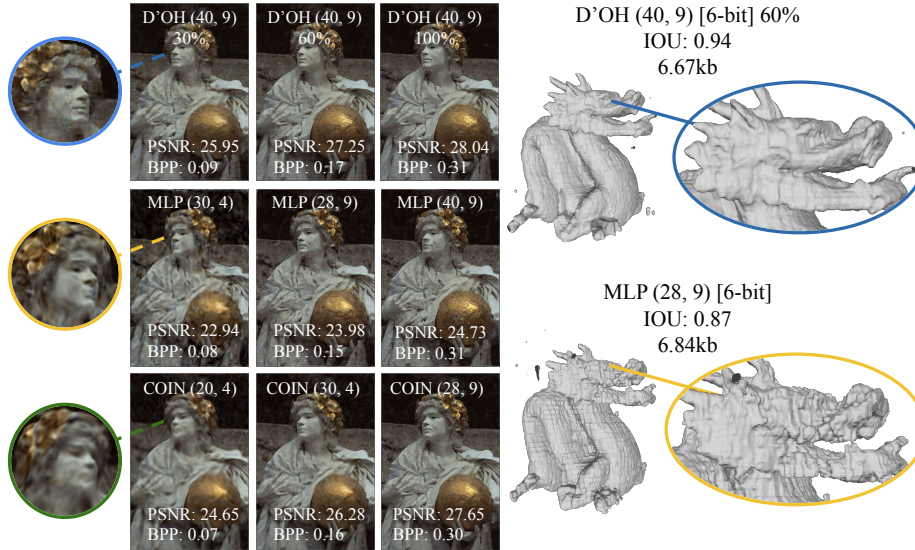


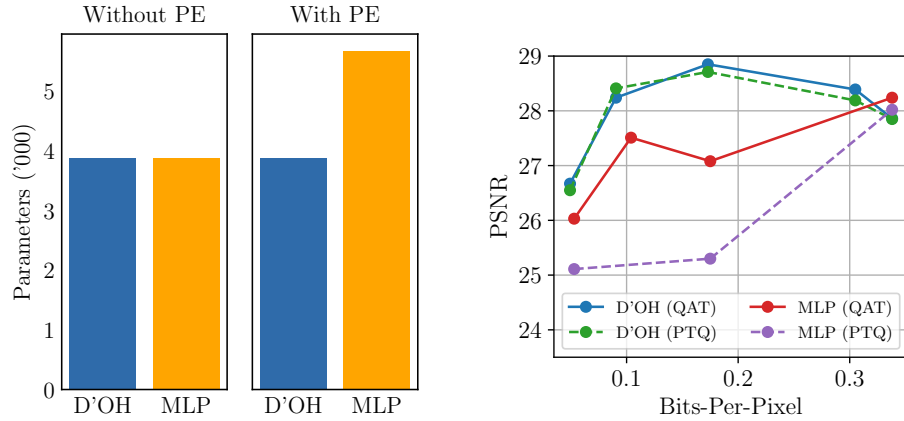
Fig. 9: Qualitative image and occupancy field results. Left: Kodak. The top row corresponds to D'OH (8-bit quantization) with a target model (40 width, 9 hidden layers) with varied latent dimension. The second and third rows are MLPs (8-bit and 16-bit [COIN]) to match approximate bit-rates. Right: Binary Occupancy Field. D'OH is robust to quantization, enabling occupancy reconstruction at low quantization levels.

representing a prediction of voxel occupancy [55]. We test the ability of D'OH to represent occupancy fields by exploring the implementation provided by [65] using Thai Statue, Dragon, and Armadillo instances [1]. The occupancy training set is constructed by sampling voxels in a coordinate lattice, with an indicator function indicating whether the voxel is filled. As in the image experiments,

we use a target network of (40, 9), and approximate this with smaller latent codes. We report performance using Intersection Over Union (IoU) \uparrow . Qualitative results and are generated by applying marching cubes across a thresholded set of sampled coordinates [50], and are shown in Figure 9. Positional encoding is required for MLP baselines to achieve suitable performance. In contrast to the image regression experiments, we find that we need to quantize to 6-bits or lower to see a clear rate-distortion improvement over MLPs with positional encoding (see: Figure 8). In the Supplementary Materials 4, we provide ablations showing the impact of quantization levels, target architectures, and positional encoding.

5 Discussion

In the previous section we showed that a Decoder-Only Hypernetwork is able to compactly represent a signal using a small latent code and demonstrate its potential for signal agnostic compression by quantizing in a manner similar to COIN [22]. Here we outline two interesting aspects of the model that occur from using a latent code and random projections to approximate a target network.



(a) For very low small networks, the change in parameters induced by positional encoding can be significant: using a 10 frequency positional encoding for a 4 layer, 30 width network increases the number of MLP parameters by 46%.

(b) Without QAT, MLPs suffer from quantization error. In contrast, due to random projection smoothing, D'OH performs similarly for both QAT and PTQ for moderate quantization levels. Kodak index 12, 8-bit quantization.

Fig. 10: The use of a random-linear hypernetwork for D'OH enables benefits beyond signal compression, including the use of positional encoding without an increase in model parameters and improved robustness to quantization without requiring QAT.

Positional Encoding The number of parameters used in the D'OH model is independent of the dimensions of the target network. An interesting consequence

of this is that while including a *positional encoding* component to a multi-layer perceptron increases the number of parameters (due to the increased input dimension), this is not true for D’OH. As a result the D’OH model is able to use positional encoding "for free". Figure 10a shows for this can be significant for small networks. We find in Supplementary Figure 3 that the increase in parameters is sufficient to lead to reduced rate-distortion performance for small MLPs on image regression with positional encoding. In contrast, for the binary occupancy experiments we find that both the MLP and D’OH models noticeably improve in rate-distortion performance when applying positional encoding.

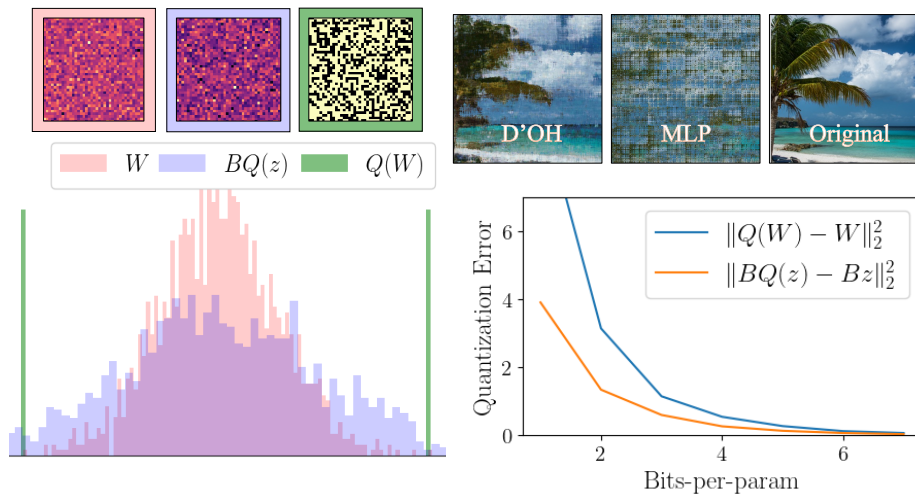


Fig. 11: Random projection helps to smooth quantization artefacts. Left: direct 1-bit quantization $Q(W)$ of a matrix W results in sharp weight discontinuities. Instead applying random projections to an quantized latent code $BQ(z)$ smooths values and better matches a unquantized distribution. Top-right: as a result D’OH can reconstruct images at 4-bit quantization, a rate which leads to catastrophic error for MLPs. Bottom-right: quantization error at different parameter quantization levels on a (40,40) matrix.

Quantization Smoothing Despite using Post-Training Quantization for our experiments we observe that D’OH has lower quantization error relative to MLPs (see: Figure 5a). This is likely due to the use of random projections in the hypernetwork. Directly applying a quantization map Q to network weights W leads to a large quantization error $\|Q(W) - W\|_2^2$ [30]. In contrast for D’OH we quantize the latent code z and *only then* apply a per-layer full-precision random mapping B to recover the target network weights. While the random projection adds no additional information it helps smooth sharp discontinuities in the weight space and allows the output distribution of $BQ(z)$ to better match the distribution of Bz . Figure 11 shows this visually - even with 1-bit quantization $BQ(z)$ can

generate a full-distribution of weight values. Intriguingly, we find that D’OH performs similarly for both Post-Training Quantization and Quantization Aware Training (see Figure 10b), a result with practical application due to increased overhead and need to select fixed bit-rates when training with QAT [31].

6 Limitations and Conclusion

Limitations Although D’OH achieves significant quality improvements at low-rate image compression tasks (see: Figure 9), we find that performance for binary occupancy fields is more limited. While D’OH is more robust at lower quantization levels (≤ 6 -bit) than MLPs with positional encoding, fully searching across architecture and quantization levels for MLPs can achieve similar rate-distortion performance to D’OH (see Supplementary Materials 4). Secondly, there remains the issue of sizing target architectures (we take the 0.6 COIN architecture as a baseline comparison). D’OH doesn’t fully remove the need to choose a target architecture, but it does provide a natural way to vary bit-rate for a fixed target architecture. In contrast, similarly varying bit-rate for quantized MLPs requires either architecture search for a completely new network or higher quantization levels (see: Figure 3). In Supplementary Figure 5 we provide an ablation of target architectures showing that varying the latent code follows a smooth Pareto frontier, with larger targets dominating smaller ones for given bit-rates, partially alleviating this concern. Finally, D’OH shares a limitation common to INRs in training time. For example, fitting a Kodak image with 2000 epochs takes approximately half-an-hour per instance. Note that [22] use 50,000 iterations per instance, and [75] up to 25,000 for their non-meta-learned models. A full dataset evaluation on Kodak (24 images) therefore takes ~ 12 h per rate configuration.

Conclusion In conclusion we have introduced a framework for direct optimization of a Decoder-Only Hypernetworks for INRs. Contrasting prior work applying hypernetworks to INRs, our method is data agnostic and does not require offline pretraining on a target signal class and is instead trained at runtime using only the target data instance. We have demonstrated the potential for a latent code and fixed random projection matrices to act as a Decoder-Only Hypernetwork, and shown that this improves upon contemporary methods for image compression such as COIN [22]. Surprisingly, this random linear hypernetwork is observed to perform similarly when trained under QAT and PTQ under moderate quantization levels which may have practical applications beyond INRs.

Acknowledgement. The authors wish to acknowledge Shin-Fang Ch’ng and Xueqian Li for their invaluable comments and discussion on the paper.

References

1. The Stanford 3D Scanning Repository. <http://graphics.stanford.edu/data/3Dscanrep/>

2. True Color Kodak Images (1991)
3. Aghajanyan, A., Gupta, S., Zettlemoyer, L.: Intrinsic Dimensionality Explains the Effectiveness of Language Model Fine-Tuning. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing. pp. 7319–7328 (2021). <https://doi.org/10.18653/v1/2021.acl-long.568>
4. Agustsson, E., Timofte, R.: NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 1122–1131 (2017). <https://doi.org/10.1109/cvprw.2017.150>
5. Alaluf, Y., Tov, O., Mokady, R., Gal, R., Bermano, A.: HyperStyle: StyleGAN Inversion with HyperNetworks for Real Image Editing. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 18490–18500 (2022). <https://doi.org/10.1109/cvpr52688.2022.01796>
6. Arora, S., Ge, R., Neyshabur, B., Zhang, Y.: Stronger Generalization Bounds for Deep Nets via a Compression Approach. In: Proceedings of the 35th International Conference on Machine Learning. pp. 254–263 (2018)
7. Ballé, J., Chou, P.A., Minnen, D., Singh, S., Johnston, N., Agustsson, E., Hwang, S.J., Toderici, G.: Nonlinear Transform Coding. *IEEE Journal of Selected Topics in Signal Processing* **15**(2), 339–353 (2021). <https://doi.org/10.1109/JSTSP.2020.3034501>
8. Barman, N., Martini, M.G., Reznik, Y.: Bjøntegaard Delta (BD): A Tutorial Overview of the Metric, Evolution, Challenges, and Recommendations (2024)
9. Bégaint, J., Racapé, F., Feltman, S., Pushparaja, A.: CompressAI: A PyTorch Library and Evaluation Platform For End-to-End Compression Research. [arXiv:2011.03029](https://arxiv.org/abs/2011.03029) (2020)
10. Bellard, F.: BPG Image format. <https://bellard.org/bpg/> (2015)
11. Bingham, E., Mannila, H.: Random Projection in Dimensionality Reduction: Applications to Image and Text Data. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 245–250. *KDD '01* (2001). <https://doi.org/10.1145/502512.502546>
12. Bjøntegaard, G.: Calculation of Average PSNR Differences Between RD-Curves. *ITU-T SG16 Q 6* (2001)
13. Chang, O., Flokas, L., Lipson, H.: Principled Weight Initialization for Hypernetworks. In: International Conference on Learning Representations (2020)
14. Chauhan, V.K., Zhou, J., Lu, P., Molaei, S., Clifton, D.A.: A Brief Review of Hypernetworks in Deep Learning. *Artificial Intelligence Review* **57**(9), 250 (2024). <https://doi.org/10.1007/s10462-024-10862-8>
15. Chen, H., He, B., Wang, H., Ren, Y., Lim, S.N., Shrivastava, A.: NeRV: Neural Representations for Videos. In: Advances in Neural Information Processing Systems. vol. 34, pp. 21557–21568 (2021)
16. Chen, Y., Wang, X.: Transformers as Meta-learners for Implicit Neural Representations. In: Computer Vision – ECCV 2022. vol. 13677, pp. 170–187 (2022). https://doi.org/10.1007/978-3-031-19790-1_11
17. Chiarlo, F.M.: Implicit Neural Representations for Image Compression. Ph.D. thesis, Politecnico Di Torino (2021)
18. Damodaran, B.B., Balcilar, M., Galpin, F., Hellier, P.: RQAT-INR: Improved Implicit Neural Image Compression. In: Data Compression Conference (DCC). pp. 208–217 (2023). <https://doi.org/10.1109/DCC55655.2023.00029>

19. Dasgupta, S., Gupta, A.: An Elementary Proof of a Theorem of Johnson and Lindenstrauss. *Random Structures & Algorithms* **22**(1), 60–65 (2003). <https://doi.org/10.1002/rsa.10073>
20. de Avila Belbute-Peres, F., Chen, Y.f., Sha, F.: HyperPINN: Learning Parameterized Differential Equations With Physics-Informed Hypernetworks. In: *The Symbiosis of Deep Learning and Differential Equations* (2021)
21. Denil, M., Shakibi, B., Dinh, L., Ranzato, M.A., de Freitas, N.: Predicting Parameters in Deep Learning. In: *Advances in Neural Information Processing Systems*. vol. 26 (2013)
22. Dupont, E., Golinski, A., Alizadeh, M., Teh, Y.W., Doucet, A.: COIN: Compression with Implicit Neural representations. In: *Neural Compression: From Information Theory to Applications – Workshop @ ICLR 2021* (2021)
23. Dupont, E., Kim, H., Eslami, S.M.A., Rezende, D.J., Rosenbaum, D.: From Data to Funct: Your Data Point is a Function and You Can Treat It Like One. In: *Proceedings of the 39th International Conference on Machine Learning*. *Proceedings of Machine Learning Research*, vol. 162, pp. 5694–5725 (2022-07-17/2022-07-23)
24. Dupont, E., Loya, H., Alizadeh, M., Golinski, A., Teh, Y.W., Doucet, A.: COIN++: Neural Compression Across Modalities. *Transactions on Machine Learning Research* (2022)
25. Elsken, T., Metzen, J.H., Hutter, F.: Neural Architecture Search: A Survey. *Journal of Machine Learning Research* **20**(55), 1–21 (2019)
26. Finn, C., Abbeel, P., Levine, S.: Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In: *Proceedings of the 34th International Conference on Machine Learning*. pp. 1126–1135 (2017)
27. Galanti, T., Wolf, L.: On the Modularity of Hypernetworks. In: *Advances in Neural Information Processing Systems*. vol. 33, pp. 10409–10419 (2020)
28. Gao, S., Huang, F., Huang, H.: Model Compression via Hyper-Structure Network (2021)
29. Garnelo, M., Rosenbaum, D., Maddison, C., Ramalho, T., Saxton, D., Shanahan, M., Teh, Y.W., Rezende, D., Eslami, S.M.A.: Conditional Neural Processes. In: *Proceedings of the 35th International Conference on Machine Learning*. *Proceedings of Machine Learning Research*, vol. 80, pp. 1704–1713 (2018-07-10/2018-07-15)
30. Gersho, A., Gray, R.M.: *Vector Quantization and Signal Compression* (1992). <https://doi.org/10.1007/978-1-4615-3626-0>
31. Gholami, A., Kim, S., Dong, Z., Yao, Z., Mahoney, M.W., Keutzer, K.: A Survey of Quantization Methods for Efficient Neural Network Inference. In: *Low-Power Computer Vision* (2022)
32. Glorot, X., Bengio, Y.: Understanding the Difficulty of Training Deep Feedforward Neural Networks. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. pp. 249–256 (2010)
33. Goodfellow, I., Bengio, Y., Courville, A.: *Deep Learning* (2016)
34. Gordon, C., Chng, S.F., MacDonald, L., Lucey, S.: On Quantizing Implicit Neural Representations. In: *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. pp. 341–350 (2023). <https://doi.org/10.1109/WACV56688.2023.00042>
35. Ha, D., Dai, A., Le, Q.V.: HyperNetworks. *International Conference on Learning Representations (ICLR)* (2017)
36. Halko, N., Martinsson, P.G., Tropp, J.A.: Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. *SIAM Review* **53**(2), 217–288 (2011). <https://doi.org/10.1137/090771806>

37. Han, S., Mao, H., Dally, W.J.: Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. *International Conference on Learning Representations (ICLR)* (2016). <https://doi.org/10.48550/arxiv.1510.00149>
38. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 770–778 (2016). <https://doi.org/10.1109/cvpr.2016.90>
39. Hinton, G., Vinyals, O., Dean, J.: Distilling the Knowledge in a Neural Network (2015). <https://doi.org/10.48550/arXiv.1503.02531>
40. Johnson, W.B., Lindenstrauss, J.: Extensions of Lipschitz mappings into Hilbert space. *Contemporary mathematics* **26**, 189–206 (1984). <https://doi.org/10.1090/conm/026/737400>
41. Karimi Mahabadi, R., Ruder, S., Dehghani, M., Henderson, J.: Parameter-efficient Multi-task Fine-tuning for Transformers via Shared Hypernetworks. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. pp. 565–576 (2021). <https://doi.org/10.18653/v1/2021.acl-long.47>
42. Kim, H., Bauer, M., Theis, L., Schwarz, J.R., Dupont, E.: C3: High-Performance and Low-Complexity Neural Compression from a Single Image or Video. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 9347–9358 (2024). <https://doi.org/10.1109/CVPR52733.2024.00893>
43. Klocek, S., Maziarka, Ł., Wolczyk, M., Tabor, J., Nowak, J., Śmieja, M.: Hypernetwork Functional Image Representation. In: *Artificial Neural Networks and Machine Learning – ICANN 2019: Workshop and Special Sessions*. pp. 496–510. *Lecture Notes in Computer Science* (2019). https://doi.org/10.1007/978-3-030-30493-5_48
44. Kopiczko, D.J., Blankevoort, T., Asano, Y.M.: VeRA: Vector-based Random Matrix Adaptation. In: *International Conference on Learning Representations* (2024)
45. Ladune, T., Philippe, P., Henry, F., Clare, G., Leguay, T.: COOL-CHIC: Coordinate-based low complexity hierarchical image codec. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. pp. 13515–13522 (2023). <https://doi.org/10.1109/iccv51070.2023.01243>
46. Lee, J., Tack, J., Lee, N., Shin, J.: Meta-Learning Sparse Implicit Neural Representations. In: *Advances in Neural Information Processing Systems*. vol. 34, pp. 11769–11780 (2021)
47. Li, X., Kaesemodel Pontes, J., Lucey, S.: Neural Scene Flow Prior. In: *Advances in Neural Information Processing Systems*. vol. 34, pp. 7838–7851 (2021)
48. Littwin, E., Galanti, T., Wolf, L., Yang, G.: On infinite-width hypernetworks. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems. Nips '20* (2020)
49. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep Learning Face Attributes in the Wild. In: *IEEE International Conference on Computer Vision (ICCV)*. pp. 3730–3738 (2015). <https://doi.org/10.1109/ICCV.2015.425>
50. Lorensen, W.E., Cline, H.E.: Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In: *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*. pp. 163–169. *SIGGRAPH '87* (1987). <https://doi.org/10.1145/37401.37422>
51. Lu, Y., Jiang, K., Levine, J.A., Berger, M.: Compressive Neural Representations of Volumetric Scalar Fields. *Computer graphics forum* **40**(3), 135–146 (2021). <https://doi.org/10.1111/cgf.14295>

52. Majumdar, R., Jadhav, V., Deodhar, A., Karande, S., Vig, L., Runkana, V.: HyperLoRA for PDEs. <https://arxiv.org/abs/2308.09290v1> (2023)
53. Martinez, J., Shewakramani, J., Wei Liu, T., Andrei Barsan, I., Zeng, W., Urtasun, R.: Permute, Quantize, and Fine-tune: Efficient Compression of Neural Networks. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 15694–15703 (2021). <https://doi.org/10.1109/cvpr46437.2021.01544>
54. McDonnell, M.D., Gong, D., Parvaneh, A., Abbasnejad, E., van den Hengel, A.: RanPAC: Random Projections and Pre-trained Models for Continual Learning. In: Advances in Neural Information Processing Systems. vol. 36, pp. 12022–12053 (2023)
55. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy Networks: Learning 3D Reconstruction in Function Space. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4455–4465 (2019). <https://doi.org/10.1109/CVPR.2019.00459>
56. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In: Computer Vision – ECCV 2020. vol. 12346, pp. 405–421 (2020). https://doi.org/10.1007/978-3-030-58452-8_24
57. Nguyen, P., Tran, T., Le, K., Gupta, S., Rana, S., Nguyen, D., Nguyen, T., Ryan, S., Venkatesh, S.: Fast Conditional Network Compression Using Bayesian Hyper-Networks. *ECML* **12977**, 330–345 (2022). https://doi.org/10.1007/978-3-030-86523-8_20
58. Nowlan, S.J., Hinton, G.E.: Simplifying Neural Networks by Soft Weight-Sharing. *Neural Computation* **4**(4), 473–493 (1992). <https://doi.org/10.1162/neco.1992.4.4.473>
59. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 165–174 (2019). <https://doi.org/10.1109/CVPR.2019.00025>
60. Pham, T., Yang, Y., Mandt, S.: Autoencoding Implicit Neural Representations for Image Compression. In: ICML Workshop Neural Compression: From Information Theory to Applications (2023)
61. Pistilli, F., Valsesia, D., Fracastoro, G., Magli, E.: Signal Compression via Neural Implicit Representations. In: ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 3733–3737 (2022). <https://doi.org/10.1109/icassp43922.2022.9747208>
62. Ramanujan, V., Wortsman, M., Kembhavi, A., Farhadi, A., Rastegari, M.: What’s Hidden in a Randomly Weighted Neural Network? In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11893–11902 (2020)
63. Ramasinghe, S., Lucey, S.: Beyond Periodicity: Towards a Unifying Framework for Activations in Coordinate-MLPs. In: Computer Vision – ECCV 2022. vol. 13693, pp. 142–158 (2022). https://doi.org/10.1007/978-3-031-19827-4_9
64. Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks. In: Computer Vision – ECCV 2016. vol. 9908, pp. 525–542 (2016). https://doi.org/10.1007/978-3-319-46493-0_32
65. Saragadam, V., LeJeune, D., Tan, J., Balakrishnan, G., Veeraraghavan, A., Baraniuk, R.G.: WIRE: Wavelet Implicit Neural Representations. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 18507–18516 (2023). <https://doi.org/10.1109/CVPR52729.2023.01775>

66. Savarese, P., Maire, M.: Learning Implicitly Recurrent CNNs Through Parameter Sharing. In: International Conference on Learning Representations (2019)
67. Schwarz, J., Teh, Y.W.: Meta-Learning Sparse Compression Networks. *Transactions on Machine Learning Research* (2022)
68. Schwarz, J.R., Tack, J., Teh, Y.W., Lee, J., Shin, J.: Modality-Agnostic Variational Compression of Implicit Neural Representations. In: Proceedings of the 40th International Conference on Machine Learning. pp. 30342–30364 (2023)
69. Sen, B., Singh, G., Agarwal, A., Agaram, R., Krishna, M., Sridhar, S.: HyP-NeRF: Learning improved NeRF priors using a HyperNetwork. In: Advances in Neural Information Processing Systems. vol. 36, pp. 51050–51064 (2023)
70. Sendera, M., Przewiezlikowski, M., Karanowski, K., Zieba, M., Tabor, J., Spurek, P.: HyperShot: Few-Shot Learning by Kernel HyperNetworks. In: IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 2468–2477 (2023). <https://doi.org/10.1109/WACV56688.2023.00250>
71. Sendera, M., Przewiezlikowski, M., Miksa, J., Rajska, M., Karanowski, K., Zięba, M., Tabor, J., Spurek, P.: The General Framework for Few-Shot Learning by Kernel HyperNetworks. *Machine Vision and Applications* **34**(4), 53 (2023). <https://doi.org/10.1007/s00138-023-01403-4>
72. Seward, J.: Bzip2 and Libbzip2. <https://sourceware.org/bzip2/> (1996)
73. Shi, J., Guillemot, C.: Distilled Low Rank Neural Radiance Field with Quantization for Light Field Compression (arXiv:2208.00164) (2022). <https://doi.org/10.48550/arXiv.2208.00164>
74. Sitzmann, V., Martel, J., Bergman, A., Lindell, D., Wetzstein, G.: Implicit Neural Representations with Periodic Activation Functions. In: Advances in Neural Information Processing Systems. vol. 33, pp. 7462–7473 (2020)
75. Strümler, Y., Postels, J., Yang, R., Gool, L.V., Tombari, F.: Implicit Neural Representations for Image Compression. In: Computer Vision – ECCV 2022. vol. 13686, pp. 74–91 (2022). https://doi.org/10.1007/978-3-031-19809-0_5
76. Szatkowski, F., Piczak, K.J., Spurek, P., Tabor, J., Trzcinski, T.: HyperSound: Generating Implicit Neural Representations of Audio Signals with Hypernetworks. In: Sixth Workshop on Meta-Learning at the Conference on Neural Information Processing Systems (2022)
77. Takikawa, T., Evans, A., Tremblay, J., Müller, T., McGuire, M., Jacobson, A., Fidler, S.: Variable Bitrate Neural Fields. In: Special Interest Group on Computer Graphics and Interactive Techniques Conference Proceedings. pp. 1–9 (2022). <https://doi.org/10.1145/3528233.3530727>
78. Takikawa, T., Müller, T., Nimier-David, M., Evans, A., Fidler, S., Jacobson, A., Keller, A.: Compact Neural Graphics Primitives with Learned Hash Probing. In: SIGGRAPH Asia 2023 Conference Papers. pp. 1–10 (2023). <https://doi.org/10.1145/3610548.3618167>
79. Tancik, M., Mildenhall, B., Wang, T., Schmidt, D., Srinivasan, P.P., Barron, J.T., Ng, R.: Learned Initializations for Optimizing Coordinate-Based Neural Representations. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2845–2854 (2021). <https://doi.org/10.1109/CVPR46437.2021.00287>
80. Ullrich, K., Meeds, E., Welling, M.: Soft Weight-Sharing for Neural Network Compression. In: International Conference on Learning Representations (2016)
81. Vonderfecht, J., Liu, F.: Predicting the Encoding Error of SIRENs. *Transactions on Machine Learning Research* (2024)

82. Wang, A.Q., Dalca, A.V., Sabuncu, M.R.: Computing Multiple Image Reconstructions with a Single Hypernetwork. *Machine Learning for Biomedical Imaging* **1**(June 2022), 1–25 (2022). <https://doi.org/10.59275/j.melba.2022-e5ec>
83. Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E.: Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing* **13**(4), 600–612 (2004). <https://doi.org/10.1109/TIP.2003.819861>
84. White, C., Safari, M., Sukthanker, R., Ru, B., Elsken, T., Zela, A., Dey, D., Hutter, F.: Neural Architecture Search: Insights from 1000 Papers (2023). <https://doi.org/10.48550/arXiv.2301.08727>
85. Wu, Q., Bauer, D., Chen, Y., Ma, K.L.: HyperINR: A Fast and Predictive Hypernetwork for Implicit Neural Representations via Knowledge Distillation (2023). <https://doi.org/10.48550/arXiv.2304.04188>
86. Xie, H., Li, J., Xue, H.: A Survey of Dimensionality Reduction Techniques Based on Random Projection (2018). <https://doi.org/10.48550/arXiv.1706.04371>
87. Xie, Y., Takikawa, T., Saito, S., Litany, O., Yan, S., Khan, N., Tombari, F., Tompkin, J., Sitzmann, V., Sridhar, S.: Neural Fields in Visual Computing and Beyond. *Computer Graphics Forum* **41**(2), 641–676 (2022). <https://doi.org/10.1111/cgf.14505>
88. Yüce, G., Ortiz-Jiménez, G., Besbinar, B., Frossard, P.: A Structured Dictionary Perspective on Implicit Neural Representations. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 19206–19216 (2022). <https://doi.org/10.1109/CVPR52688.2022.01863>
89. Zhang, C., Ren, M., Urtasun, R.: Graph HyperNetworks for Neural Architecture Search. In: *International Conference on Learning Representations (2019)* (2019)
90. Zhang, G., Zhang, X., Tang, L.: Enhanced Quantified Local Implicit Neural Representation for Image Compression. *IEEE Signal Processing Letters* **30**, 1742–1746 (2023). <https://doi.org/10.1109/lsp.2023.3334956>
91. Zhang, K., Zhu, D., Min, X., Zhai, G.: Implicit Neural Representation Learning for Hyperspectral Image Super-Resolution. In: *IEEE International Conference on Multimedia and Expo (ICME)*. pp. 1–6 (2022). <https://doi.org/10.1109/ICME52920.2022.9859739>
92. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 586–595 (2018). <https://doi.org/10.1109/CVPR.2018.00068>