Implicit-Zoo: A Large-Scale Dataset of Neural Implicit Functions for 2D Images and 3D Scenes

Qi Ma[1,2] **Danda Pani Paudel**[2] **Ender Konukoglu**[1] **Luc Van Gool**[1,2]
[1]Computer Vision Lab, ETH Zurich [2]INSAIT, Sofia University

Abstract

Neural implicit functions have demonstrated significant importance in various areas such as computer vision, graphics. Their advantages include the ability to represent complex shapes and scenes with high fidelity, smooth interpolation capabilities, and continuous representations. Despite these benefits, the development and analysis of implicit functions have been limited by the lack of comprehensive datasets and the substantial computational resources required for their implementation and evaluation. To address these challenges, we introduce "Implicit-Zoo": a large-scale dataset requiring thousands of GPU training days designed to facilitate research and development in this field. Our dataset includes diverse 2D and 3D scenes, such as CIFAR-10, ImageNet-1K, and Cityscapes for 2D image tasks, and the OmniObject3D dataset for 3D vision tasks. We ensure high quality through strict checks, refining or filtering out low-quality data. Using Implicit-Zoo, we showcase two immediate benefits as it enables to: (1) *learn token locations* for transformer models; (2) *directly regress* 3D cameras poses of 2D images with respect to NeRF models. This in turn leads to an *improved performance* in all three task of image classification, semantic segmentation, and 3D pose regression – thereby unlocking new avenues for research. Our data and implementation are available from: https://github.com/qimaqi/Implicit-Zoo/

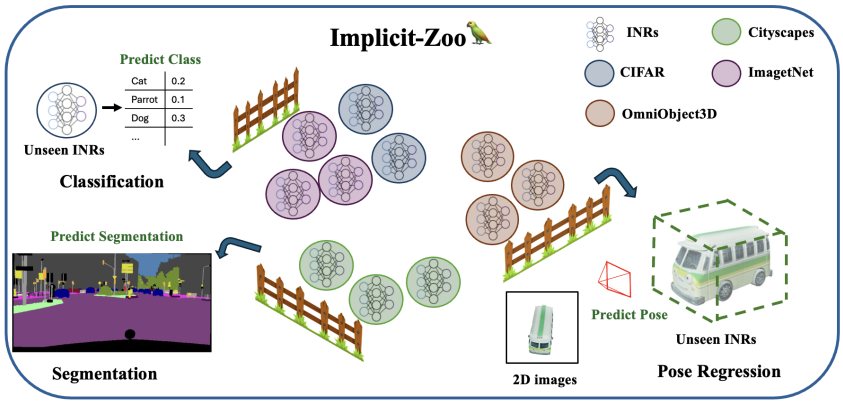


Figure 1: ***The Implicit-Zoo dataset and its example utilities.*** We demonstrate three tasks using *Implicit-Zoo*: classification, segmentation, and 3D pose regression. Details of the problem statement can be found in section 4. The INRs are colorized differently to indicate their training data sources.

38th Conference on Neural Information Processing Systems (NeurIPS 2024) Track on Datasets and Benchmarks.

Method	Task	Scenes	Model(Depth/Width)	GPU (days)	Overall Size (GB)	PSNR
CIFAR-10 [1]	2D	60000	3 / 64	5.96	1.44	31.01
ImageNet-1K [2]	2D	1431167	4 / 256	831.53	749.93	30.12
CityScapes [3]	2D	23473	5 / 256	50.15	18.40	34.10
Omniobject-3D [4]	3D	5914	4 / 128	69.81	5.96	31.54

Table 1: **Dataset Summary.** An overview of the generated datasets with their cost and quality. For the 2D task, we employ SIREN [5], while for the 3D task, we utilize NERF [6]. Computation is carried out on the ETH Euler cluster. We report PSNR (Peak Signal-to-Noise Ratio) as quality metric. A PSNR of 30 dB corresponds to an RGB MSE of appr. 0.03. This error level is hardly noticeable to the human eye, indicating the high fidelity in our dataset. Please refer to Figure 2 for examples.

1 Introduction

Recent advances in modeling continuous functions implicitly using Multi-Layer Perceptrons (MLPs) [7] have sparked great interest in many applications. Implicit neural representations (INRs) involve fitting a function $f(.)$ that maps input coordinates x to their corresponding value v_x. For instance, in image modeling [5], this continuous function maps 2D pixel coordinates to RGB values. This approach has been extended to challenging 3D geometry and appearance [6, 8–10]. Implicit representations offer many advantages, including strong modeling effectiveness, memory efficiency, the ability of a single architecture to represent different data modalities, compatibility with arbitrary resolutions, and differentiability.

A hindrance to further advancements in implicit function research is the lack of large-scale datasets, primarily due to their computational demands. This paper aims to bridge this gap via the *Implicit-Zoo* Datasets, with access to over 1.5 million implicit functions across multiple 2D and 3D tasks.

Some INRs datasets exist in both 2D and 3D formats. However, they are limited by data scale [11] and application scenarios [12, 13]. Additionally, while using modulation [14, 15] to learn the common parts of given dataset to accelerate convergence, it struggles to effectively handle unseen data samples. We will also organize and update the datasets used for concurrent work. For example, [16] use Instant-NGP [17] to train a large amount of indoor data.

We focus on generating dataset using INRs directly on modelling image signals because of their wide-ranging applications, following the existing remarkable success [18, 19]. We believe our dataset will have broad applicability and make a significant contribution to the community. Additionally, we chose popular 2D datasets because they have well-established benchmarks [2, 3, 1], enabling better performance comparison with other state-of-the-art algorithms.

Moreover, we developed a comprehensive 3D Implicit Neural Representations (INRs) dataset using Ombiobject3D [4] and establish the first benchmark for 3D INRs pose regression. As INRs gain the status of preferred representation for 3D scenes, determining pose from given images with trained INRs becomes crucial. [20–22] aim to jointly address reconstruction and image registration challenges during the training phase of Implicit Neural Representations (INRs) without relying on pose priors.[23–25] focus on pose regression using pretrained INRs. However, these methods only achieve convergence in scenarios with coarse pose initialization or when a scene-dependent regressor is trained, limiting their applicability to new scenes. To address this, we introduce a transformer-based approach that samples the neural radiance field to extract volume feature, integrating it with 2D images for precise pose regression. In unseen scenarios, our method achieves a rotational error of 20^o, with nearly 80% of poses having rotational errors below 30^o. Further refinement is shown by minimizing a photometric error [20].

To effectively train our large-scale dataset, we carefully selected model sizes 1 appropriate for the complexity of the data and ran a sufficient number of iterations. To maintain data quality, we conducted a second training round to guarantee that all images reached a PSNR of 30.

Thanks to the differentiability of INRs and our large-scale dataset, our transformer-based methods can efficiently optimize tokenization for various tasks. Instead of using manually designed tokenization techniques like standard patchification or volumification, our approach allows the network to learn the tokenization process directly from the large-scale dataset. We propose methods that utilize learnable

patch centers and scales, as well as a fully learnable approach at the pixel-wise or point-wise level. Our findings indicate that this learnable tokenization significantly improves performance across multiple tasks. This research uncovers a novel direction: learnable tokenization.

In summary: our key contributions are as follows:
• We create *Implicit-Zoo*, a large-scale implicit function dataset developed over almost 1000 GPU days. Through iterative refinement, incl. filtering and continuous training, we ensure its high quality.
• We benchmark a range of tasks using this dataset, such as 2D image classification and segmentation. Additionally, we introduce a transformer-based approach for the direct pose regression for new images in the 3D neural radiance fields. For the latter, a novel baseline is also established.

• By integrating learnable tokenizer, we enhance the benchmark methods across multiple tasks.

2 Related Work

2.1 Implicit neural representations (INRs)

SIREN [5] use periodic activation functions to capture high-frequency details in images. Building on this, [26] propose using Gaussian functions, which offer greater robustness to random initialization and require fewer parameters. [27] introduce a continuous complex Gabor wavelet to robustly represent natural images with high accuracy. [28] focus on continuous representation for arbitrary resolution.Implicit Neural Representations (INRs) have been shown to effectively represent scenes as occupancy [8, 27, 5], object shape [29, 9, 30], sign distance function [10, 31], 3D scene appearance and dynamics [6, 7, 32–37] and other complex signals and problems[38–42].

2.2 Transformer on various computer vision tasks

Vision Transformers (ViTs) have achieved state-of-the-art (SOTA) performance in several **Image recognition** tasks[43] and are thus selected as our primary benchmark. It rely on the self-attention mechanism [44, 45] and require large datasets for effective training. This underscores the need for large datasets in Implicit Neural Representations (INRs) format, especially for transformer-based methods. Typically, after pre-training in either a supervised [18, 46] or self-supervised [47–51] manner, fine-tuning is performed on smaller datasets for downstream tasks. **Image segmentation,** which involves dense prediction and requires a larger number of tokens, poses challenges for ViTs with fixed token number and dimension. [52–57, 52, 58, 59] addressed this issue by hierarchical reducing token numbers and increase token feature dimension through convolution backend. Such multi-stage design perform well for also recognition task.[19] developed a lightweight transformer encoder that incorporates sequence reduction with a lightweight MLP decoder. In the field of **pose regression,** where transformer-based methods have also shown success[60–66]. Many of them build upon [53] which outputs a set of tuples with fixed cardinality for detection tasks.

2.3 Tokenizer of Transformer

Recent advancements in vision transformers have focused on improving tokenization strategies to enhance performance and efficiency. [44] explored the effectiveness of tokenizers through Modulation across Tokens (MoTo) and TokenProp. The T2T-ViT model by [18] introduced progressive tokenization and an efficient backbone inspired by CNNs, leading to significant performance gains on ImageNet. [26] proposed MSViT, a dynamic mixed-scale tokenization scheme that adapts token scales based on image content, optimizing the accuracy-complexity trade-off. [47] developed token labeling, a dense supervision approach that reformulates image classification into token-level recognition tasks, greatly enhancing ViT performance and generalization on downstream tasks.

2.4 Pose regression within Neural Radiance Field

LENS[25] applies novel view synthesis to the robot re-localization problem, using NeRF to render synthetic datasets that improve camera pose regression. Loc-NeRF [23] combines Monte Carlo localization with NeRF to perform real-time robot localization using only an RGB camera. iNeRF [24]inverts a trained NeRF to estimate 6DoF poses through gradient descent and propose proper ray sampling mechanism. Note that all method need heavy volumetric rendering process.

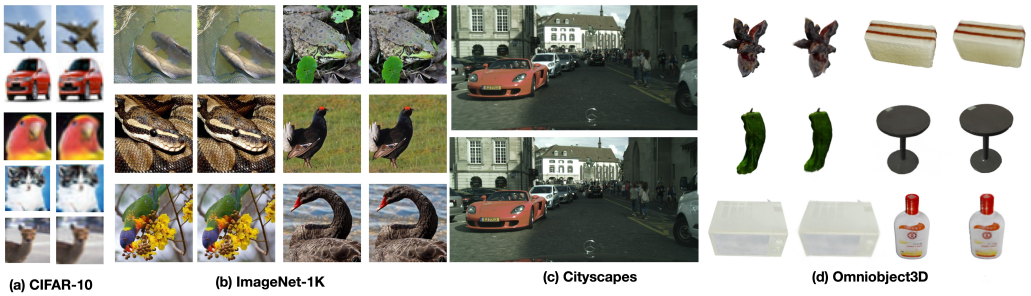


Figure 2: **Examples of images from INRs.** We present visual comparisons of example image pairs from our Implicit-Zoo dataset. The original (left/top) and the reconstruction from INRs (right/bottom) images are presented in pairs, showcasing similar visual quality. Please zoom in for details.

3 The Implicit-Zoo Dataset

We introduce the *Implicit-Zoo* dataset which includes over 1.5 million INRs and took almost 1000 GPU days for training on RTX-2080. We first describe the dataset over different tasks and generation process, incl. some necessary changes and the cost for large scale training of implicit functions. Then we go though the quality check and lastly we describe the licence for the released dataset.

3.1 Dataset generation

CIFAR-10-INRs [1] We test our algorithm with CIFAR-10. We employ a 3-layer SIREN MLP with a width of 64, without positional encoding [5]. For all 2D image tasks the images are normalized using a mean of $[0.485, 0.456, 0.406]$ and a standard deviation of $[0.229, 0.224, 0.225]$. We use the Adam optimizer with a learning rate of 1e-3 and a cosine learning rate scheduler with minimum learning 1e-5 without warmup. Each image is trained for 1000 iterations, requiring 8.58 secs in total.

ImageNet-1K-INRs [2] Following the standard procedure for ImageNet-1K classification, we resize images to 256x256 and then center crop them to 224x224 for further processing. To better fit the high-res images, we employ a 4-layer 256 width SIREN. The normalization parameters are consistent with those used for CIFAR-10. We increase the training iterations to 3000, each taking 50.24 secs.

CityScapes-INRs [3] The Cityscapes dataset focuses on semantic understanding of urban street scenes, requiring exceptionally high image quality for pixel-wise classification. We resize the images from 1024x2048 to 320x640 pixels. To handle the details, we use a 5-layer SIREN model with a width of 256 and 3000 training iterations. The cost for training individual INR is 184.3 secs.

Omniobject3D-INRs [4] The Omniobject3D dataset comprises 5998 objects across 190 daily categories. For training, we utilize the official implementation of NeRF[6], assuming a white background. Our setup includes a range of [2, 6], 64 samples per ray, and 2048 rays in one batch. Initially, we resize the images from 800x800 to 400x400 pixels and use first 96 views to generate the neural radiance field. The time cost for training a single scene is 1019 seconds.

3.2 Dataset quality control

To account for varying convergence rates across images and to optimize time efficiency, we have developed a third-phase training framework. First, we conduct basic training with a predefined number of iterations and a learning rate scheduler. Second, for data that fails to achieve a PSNR of 30 dB after basic training, we initiate extended training, which allows for up to three times the number of iterations as the basic phase. An early-stopping mechanism is incorporated into the extended training to prevent unnecessary computation.In addition, we conducted a thorough data check after, performing further training on all data that still had not achieved a PSNR of 30. For this round we train till the PSNR threshold is met. Some examples of data is shown in Fig 2.

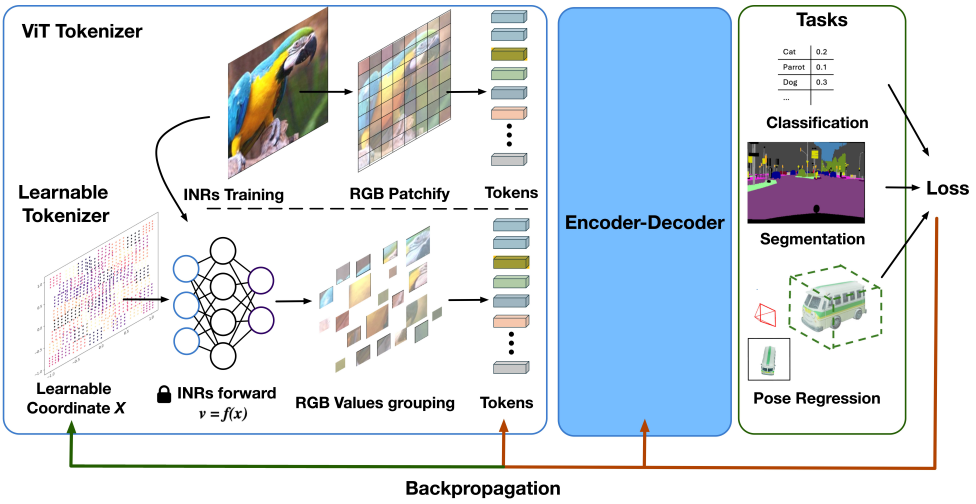


Figure 3: **Illustration of learnable tokenizer.** Instead of retrieve RGB value from images we query learnable coordinates to pre-trained freezed INRs and grouping RGB values to create tokens. Note that during backpropogation the Coordinate x will also be jointly optimized with ViT modules.

.

3.3 Data protection and licence

Implicit-Zoo is constructed using several popular RGB datasets. **CIFAR10**: MIT-License, we are allowed to redistribute under same terms and we will release it on Kaggle. **ImageNet-1K**: Similar to CIFAR we will redistribute INRs on Kaggle, consistent with the original distribution protocols. **Cityscapes**: Non-commercial purposes licence, This dataset will be hosted on the Cityscapes team's official website,where users must agree to the specified terms of use. **Omniobject3D**: CC BY 4.0 licence, we are permitted to redistribute it in our format directly under the same terms.

4 Dataset Applications

4.1 Different tasks statement

Classification: We provide a dataset of INRs $\{f_i\}_{i=1}^N$ and associated labels $\{y_i\}_{i=1}^N$, $y_i \in \{0, 1, 2...C-1\}$. C is the total classes number and N is the size of dataset. The goal is to learn a model $g : f_i \rightarrow y_i$ that accurately predicts the label y_i for a given INRs f_i. Recall, the RGB values at the 2d coordinates x is obtained using INRs such that $v_x = f_i(x)$.

Segmentation: Similar to classification, in segmentation the model needs to predict densely the pixel-wise labels. To achieve this, coordinate querying is crucial for establishing pixel-wise correspondence.

Pose Regression: In pose regression, we localize an image I with respect to the 3D scene represented by its INR $f(.)$. To do so, we train a neural network $g : (f, I) \rightarrow \theta$, where θ represents the 6D pose parameters. In other words, we directly regress the pose of the image I using a neural network that leverages the INRs. Thanks to the scale of our dataset, we can train such network which also generalizes beyond the training scenes. Furthermore, our dataset allows to meaningfully learn the tokens' locations. We refer this process as learnable organization, which is presented below.

4.2 Learnable tokenizer

Vision Transformers treat input as sequences of patches [18, 54] or volumes [67] – which are also referred as tokens – from *fixed and handcrafted locations* x. Let a token be t and $z = (x, v_x)$ be a tuple of location and the corresponding value. Then for a set $\mathcal{Z} = \{z_k\}_{k=1}^M$ of M tuples, we create the token t by using a learnable function $T(.)$ such that $t = T(\mathcal{Z})$. It is important to note that the *INRs make the location x learnable with respect to token t*. Thanks to the differentiability and scalability of our Dataset, we propose to jointly optimize tokenizer by making x learnable with other ViT modules as shown in Fig 4. In our experiments, we use convolution operation $T(.)$ as tokenizer. To make this

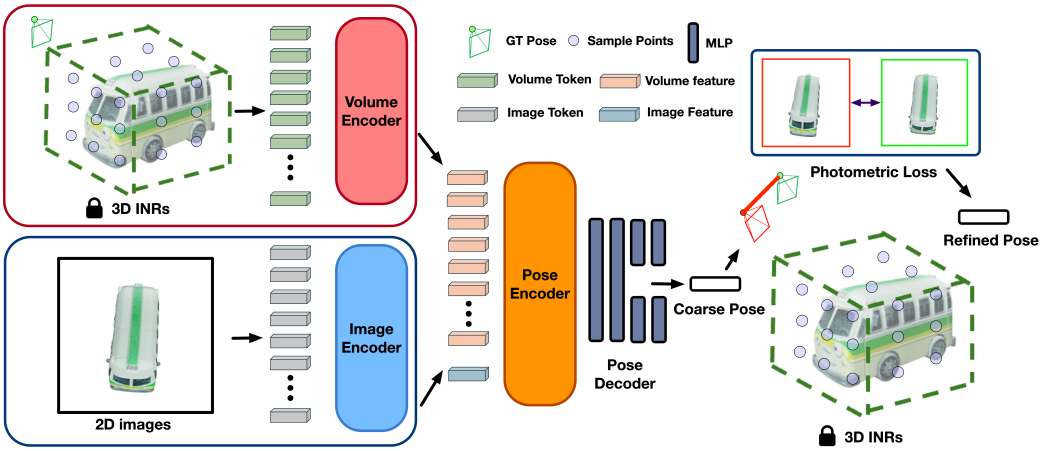
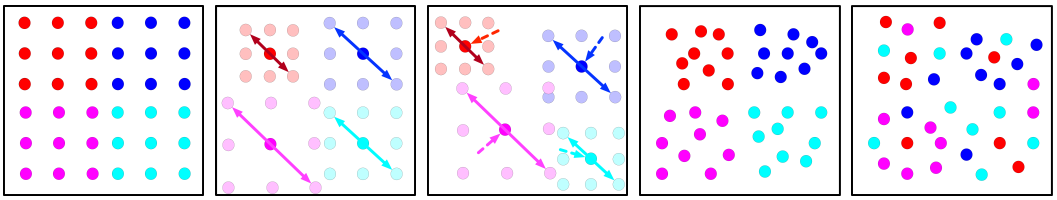


Figure 4: **Illustration of proposed pose regressor.** We process 3D volume features and 2D image features with transformer-based encoder and output coarse poses. For further refinement, we freeze the 3D INRs and optimize the pose by minimizing the photometric error.

approach effective is challenging. It requires specific *RGB grouping methods* for both 2D and 3D INRs, *ensuring differentiability* at every step especially for data augmentation. Please refer to the supplementary for details of our differentiable augmentation scheme.

4.2.1 RGB grouping strategies

ViT use uniform patches of the same size, as shown in Fig. 7a. In contrast, we propose learnable scaling and learnable center location to handle multi-resolution and allow patches to more important area, shown in Fig 5b,5c. Furthermore we propose all pixels coordinate are learnable 5d. All above mentioned coordinates are initialized with the original RGB pixel coordinates. We also investigate the random initialized shown in Fig and 5e. Corresponding quantitative and qualitative results please refer to 6 and 6.



(a) Uniform Patchify (b) Uniform Centers + Learnable Scaling (c) Learnable Centers + Learnable Scaling (d) Learnable Pixels (e) Random Initialize + Learnable Pixels

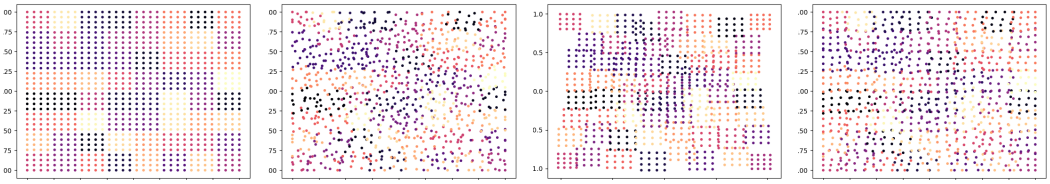
Figure 5: **Different RGB grouping strategies.** We visualize the proposed RGB grouping strategies with patch size 3. Coordinates with same color will be grouped into the same token. We abbreviate these approaches as (b) “S", (c) “LC", (d) “LP", and (e) “LP+rand".

5 The Implicit-Zoo Benchmark and Experimental Results

We report four benchmarks on the Implicit-Zoo Dataset, covering image classification, semantic segmentation, and 3D pose regression tasks. Our focus is on the RGB input space. Additional investigations on methods that use only the weight space and also training details can be found in the supplementary materials.

Method	Acc ↑	Precision ↑	F1 ↑
ViT[18]	80.82±0.86%	80.76± 0.87 %	80.75± 0.86 %
ViT[18] + S	80.24± 0.47%	80.49± 0.63%	80.44± 0.57%
ViT[18] + LC	81.33± 0.23%	81.29± 0.22%	81.30± 0.23%
ViT[18] + LP + rand	59.43 ± 1.21 %	59.56 ±1.32 %	59.65 ±1.29%
ViT[18] + LP	79.51± 0.23%	79.37± 0.34%	79.37± 0.35%
ViT[18] + LP + Reg	**81.57± 0.29%**	**81.53± 0.30%**	**81.51± 0.30%**

Table 2: **Cifar-10-INRs Classification task.** Classification results on our CIFAR-INR dataset with different grouping methods. Abbreviations are introduced in Figure 5. All results are averaged over 5 runs. Baseline results align with results reported using CIFAR-10 images.



(a) Uniform Patchify (b) Uniform Centers + Learnable Scaling (c) Learnable Centers + Learnable Scaling (d) Learnable Pixels

Figure 6: **Learned tokens in CIFAR-10 INRs settings.** Different strategies corresponding to Figure 5. Same color (chosen randomly for visualization) indicates grouping into the same token.

5.1 Benchmark methods

We choose Vision Transformer and its variants as main method for its state of the art performance across multiple tasks and to demonstrate the effectiveness of the learnable tokenizer. For classification we choose [18] as baseline method and for segmentation we use Segformer[19] for benchmark approach for its efficiency and light-weight design. For pose regression we use method shown in 4.

5.2 Benchmark experiments and results

CIFAR-10-INRs Classification: We train ViT classifier from scratch with batch size 512 for 200 epochs. We compare five different grouping methods with the baseline. In addition to the approaches mentioned in Sec. 4.2.1, we found that if all pixels are allowed to move without any constraints, multiple pixels may converge to the closing location due to local minima shown in Fig.6b. To address this, we introduced a regularization term to penalize too close coordinates within the same token t as shown in Eqn.1. We abbreviate it as "LP+reg". For N coordinates $\{x_i\}_{i=1}^{N}$ within that token, we use $\mathcal{L}1$ gate loss and choose the threshold α to be $min(1/H, 1/W)$ where H, W is the height and width of input images, such that,

$$\mathcal{L}_{reg} = \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} \text{ReLU}\left(\alpha - \|x_i - x_j\|_2\right). \tag{1}$$

Implementing this approach reveals that the proposed “LC" and “LP+reg" methods surprisingly outperform the baseline by 0.51 % and 0.75 % in accuracy. Conversely, the "LP+rand" method performs poorly, which is consistent with our expectations due to the loss of local geometry and

Method	Params	GFLOPs	mIOU ↑(fine)	mIOU ↑(coarse)
MiT-B0[19] MiT-B0	3.7	31.5	39.95 ±0.9	42.67 ±0.8
MiT-B0[19]+LC	3.7	112.5	40.33±0.7	42.70 ±0.6
MiT-B0[19]+LP+Reg	4.0	112.5	**40.61±0.6**	**43.29 ±0.3**

Table 3: **CityScapes-F[3] Segmentation.** Results obtained on the Cityscapes fine annotation dataset.

appearance information [68]. As illustrated in Fig. 6c, boundary patches tend to move towards the center, while center patches tend to overlap. With Non-Close regularization, the coordinates, as shown in Fig. 6d, retain a grid structure. This helps preserve local geometry information while leave patches the flexibility to grow and overlap.

ImageNet-100 INRs Classification: We conduct fine-tuning experiment similar to [18] and use the best RGB grouping approaches from previous experiments. After fine-tuning 8000 steps based on re-finetuned from augreg 21k-1k[68], we report similar performance improve in accuracy in Tab. 4, It is minimal potentially because the pre-trained ViT is based on fixed patches. Additionally, "LC" performs better "LP", which could also be attributed to the pre-training on grid-like tokenization.

Method	Acc ↑	Pre ↑	F1 ↑
ViT[18]	84.81±0.92 %	85.2±0.87 %	84.58±1.05 %
ViT[18]+LC	**85.02±1.02** %	85.28±1.01 %	84.64±1.03 %
ViT[18]+LP+Reg	84.92±0.93 %	**85.30±0.91** %	**84.74±1.05%**

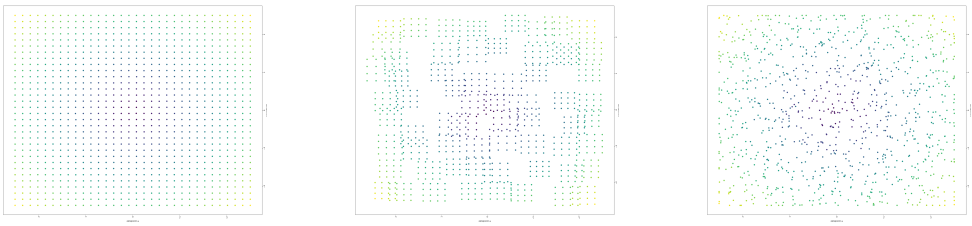
Table 4: **ImageNet-100[69] Classification task.** Fine-tuning experiments on the ImageNet-100 dataset yield results similar to those observed in the CIFAR-10 experiment 6. The improved performance is attributed to the proposed learnable tokenization, thanks to the Implicit-Zoo dataset.

CityScapes-INRs Semantic Segmentation: Follow experiments in [19] which use pretrained encoder MiT-B0 on ImageNet-1K and then fine-tune it on Cityscapes-INRs for 40k steps with batch size 16. We report also the Params and GFLOPs for using our learnable tokenizers. Note that optimizing tokenizer ffor all coordinates in high-resolution images can be very computationally expensive in terms of GFLOPs. We report baselines results similar to [19]. Surprisingly, using learnable tokenizer also improves performance in pixel-to-pixel tasks like segmentation. The misalignment between input pixels and the supervised pixels with labels can be can be resolved by tokenization and self-attention mechanisms, which help to locally aggregate relevant information.

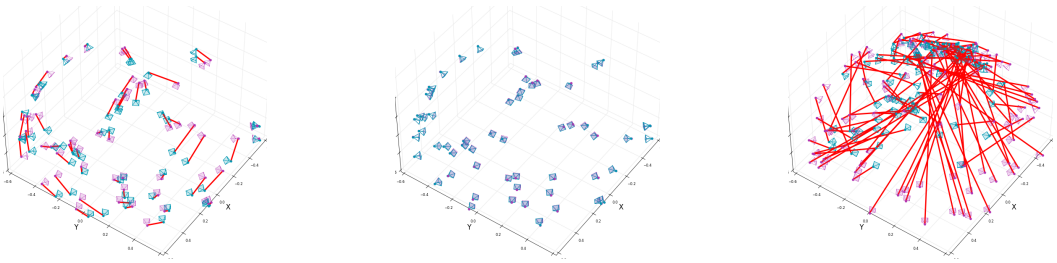
Metric	Ours	Ours+Pre	Ours+LC	Ours+Pre+LC	Ours+LP	Ours+Pre+LP
Seen Scenes						
TE(cm) ↓	3.50	3.13	3.30	3.12	3.22	**2.99**
RE ↓	15.29 °	14.40 °	15.67 °	14.59 °	14.67°	**14.17**°
RE@5 ↑	44.36%	47.30%	43.55%	47.34 %	45.11%	**50.02 %**
RE@15 ↑	75.30 %	76.50%	74.95 %	76.41%	75.21 %	**76.60 %**
RE@30 ↑	83.35%	83.85 %	83.27 %	83.75 %	83.44 %	**83.95 %**
RE+Ref ↓	6.93 °	4.35 °	7.13 °	4.22 °	4.07 °	**3.91** °
Unseen Scenes						
TE(cm) ↓	6.91	6.61	6.92	**5.96**	6.86	5.99
RE ↓	21.90 °	20.24 °	21.83 °	20.09 °	21.68 °	**20.02** °
RE@5 ↑	27.46%	29.94 %	27.46%	**30.77%**	27.97%	30.19%
RE@15 ↑	64.19 %	66.65%	64.20%	**67.07%**	64.23 %	66.9 %
RE@30 ↑	77.50 %	79.42 %	77.40%	79.44%	79.61%	**79.75%**
RE+Ref ↓	9.86 °	**9.07** °	9.85 °	9.70 °	9.01 °	**8.09** °

Table 5: **OmniObject3D-INRs pose regression.** We present pose regression results for both seen and unseen scenes in our Omniobject3D dataset. 'Pre' denotes a pretrained encoder, 'Ref' refers to further refinement. Our proposed learnable tokenization achieves better pose regression results.

Omniobject3D-INRs pose regression: Our proposed small model discretize scenes to a 32x32x32 volume and volume points are queried to and get rgb and density. We then concatenate them together and tokenize it to volume token with 3D convolutions shown in 4.We compute the $\mathcal{L}2$ distance for translation and for rotational error we use shortest rotation distance. Moreover we also report RE@β where $\beta \in (5, 15, 30)$ which indicated the percentage of rotational error below a threshold β. Our findings show that pre-trained volume encoder improves results across all methods with different grouping strategies. Moreover, optimizing the 3D tokenizer yields better performance. Although the improvements are modest, qualitative images 7b reveal that the learned patches exhibit large overlap and leave blank areas. This suggests the potential to reduce the number of patches. Additionally, our method identifies coarse positions close to the ground truth pose, which can be further optimized



(a) Uniform Volumize (b) Learnable Volume Center + Learnable Scale (c) Learnable Points



(d) Coarse Pose (e) Refined Pose (f) Failure case

Figure 7: **3D learnable token and pose regression.** We visualize the center slices of the learnable 3D tokenizer (top), where the colors indicate initialized positions. The coarse poses are directly regressed using the network trained on the Implicit-Zoo dataset. Red lines show the translation errors.

using other pose regression methods [20, 24]. More details regarding model and pre-training can be find in supplementary.

6 Conclusion and Limitations

In this paper, we introduced *Implicit-Zoo*, a large-scale implicit function dataset developed over nearly 1000 GPU days. Through strict data quality check, we have ensured its high quality. Using this dataset, we benchmarked a range of tasks, including 2D classification and segmentation. Additionally, we proposed a transformer-based approach for pose regression utilizing trained 3D neural radiance fields. By incorporating a learnable tokenizer, we enhanced the benchmark methods and discovered valuable insights for future in tokenizer research. Our work has few limitations: Scalability is restricted due to INR querying, resulting in small batch and model sizes in benchmark experiments, which limits from-scratch training of complex models. We set a PSNR threshold of 30 for dataset expansion, but this may cause artifacts in repetitive backgrounds, thus requires additional data refinement. Lastly, our pose regression task struggles with symmetric objects (Fig. 7f); this may be addressed using symmetry-aware representations [70]. We believe our dataset and its showcased applications open up new avenues for the future research.

7 Supplementary Material

In this supplementary material, we first detail the data generation process in Section 7.1 and provide more information on the implementation of the learnable tokenizer in Section 7.2. Next, we present additional details and experimental results on the benchmarks in Section 7.3. Finally, we provide information needed in checklist in Section 7.4. To gain a better understanding of our dataset and proposed benchmarks, please refer to the introductory video in the supplementary materials or the one available on our project page, which offers an overview of our dataset and its applications.

Figure 8: **Additional examples on CIFAR-10-INRs dataset** We present additional data examples, where the left side of each image pair shows the ground truth and the right side displays the results queried from the INRs.

7.1 Additional details of Dataset Generation

Speed up Training In [14], the authors propose meta-learning and implicit function modulation to accelerate the training process. Similarly, [71] and [12] reduce training time by employing smaller models and optimizing the number of iterations. In our 2D dataset, we observed that normalizing images before training implicit functions significantly speeds up training iterations. Therefore, we chose to normalize images and get rid of Sigmoid activation function in the final layer. In the 3D cases, we use a small model with 4 layers and a width of 128, without any skip connections. During training to enhance the performance with limited iterations, we propose an adaptive sampling method that focuses more on rays corresponding to 2D RGB values that are not white (likely to be the background). This approach is particularly beneficial for handling light-colored cases and tiny objects as shown in Fig 11. We observed that the training loss converged at 20k steps with learning rate 5e-4.

More Examples of data We provide more examples of data on 8, 9, 8, 11. Note that if you zoom in, you may notice some artifacts in the CIFAR-10 dataset. For example, in the bottom row of the dog category in Fig 8 , the dogs appear slightly blurry. For a 32x32 image, a PSNR of 30 results in more noticeable visual differences compared to larger images, as seen in Fig 9. To address this issue, we refined the CIFAR-10 data as described in the main paper, increasing the average PSNR to approximately 35 and resulting in a smaller standard deviation across different classes, as shown in Fig 16. Additional experiments on the refined dataset, reported in Table 7, align with the findings in the main paper.

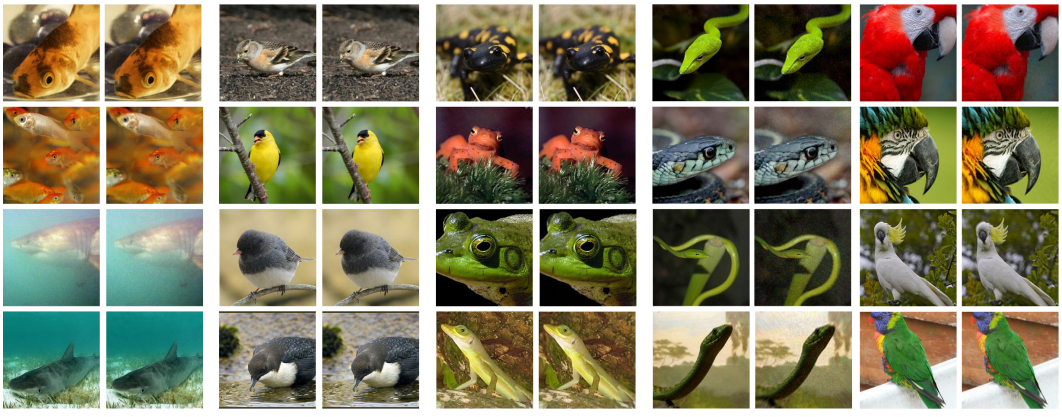


Figure 9: **Additional examples on ImageNet dataset** We present additional animal images from the ImageNet dataset, which is one of the motivations behind naming this work *Implicit Zoo*. Comparing with the ground truth images on the left, the reconstructions are of very high quality.

Data statistics We report PSNR across classes of dataset in Fig 16, 17. For Cityscapes-INRs results please refer to main page. Note that the PSNR differences in 2D cases are minimal due to the quality control and further refinement we implemented. Some of the classes in ImageNet results are higher than others, indicating better performance achieved during the initial phase of training.

Figure 10: **Additional examples on Cityscapes dataset** We present additional data samples from Cityscapes-INRs. Notably, fine details such as pedestrians in (c) and significant illumination changes in (d) are well-preserved in the reconstructions.

Scene filtering As shown in Fig 18 the rendering PSNR for novel view changes a lot cross different classes (180 claases) with standard deviation 3.87. This is mainly because the various objects the dataset include. As shown in Fig 11, we observe that PSNR tends to be higher for light-colored objects because their colors align with the white-background assumption [6]. A similar trend is observed for small objects. During data filtering, we first exclude cases with a PSNR below 25. For classes with fewer than five scenes, we ignore the entire scene in this class. Ultimately, we retain 5,287 valid scenes for our experiments.

Data releasing. We uploaded CIFAR-10-INRs, ImageNet-10-INRs Omniobject3D to Kaggle and can be found in project page. The Cityscapes-INRs dataset will be released shortly on the Cityscapes team's official after this paper is published publicly. Additionally, we are working on a refined version of ImageNet with PSNR > 35 and training a larger NeRF model on Omniobject3D, utilizing a coarse and fine model with 8 layers and a width of 256. The benchmark code will also be released on the above webpage.

Figure 11: **Additional examples on Omniobject3D dataset** We present additional examples on Omniobjecct3D. We observe that when objects are large, have rich colors, and relatively simple surfaces, our reconstruction performs very well (a). However, in more challenging cases such as (b) shallow-colored objects, (c) complex surfaces with text information, and (d) small or thin objects, the reconstruction quality is less satisfactory.

7.2 Additional details of Learnable tokenizer

2D implementation We provide more detailed information on learnable tokenizer and different RGB grouping on 2D implementation here. We first map the coordinates to $(-1, 1)$ and then divide them uniformly with patch size P to N patches, each containing P^2 coordinates. We calculate the center coordinate c_i for each patch $i \in \{1, 2, ...N\}$ and determine the coordinate difference of each coordinates to the center coordinate d_{ij}, where $j \in \{1, 2, ...P^2\}$. Thus, all coordinates x_{ij}

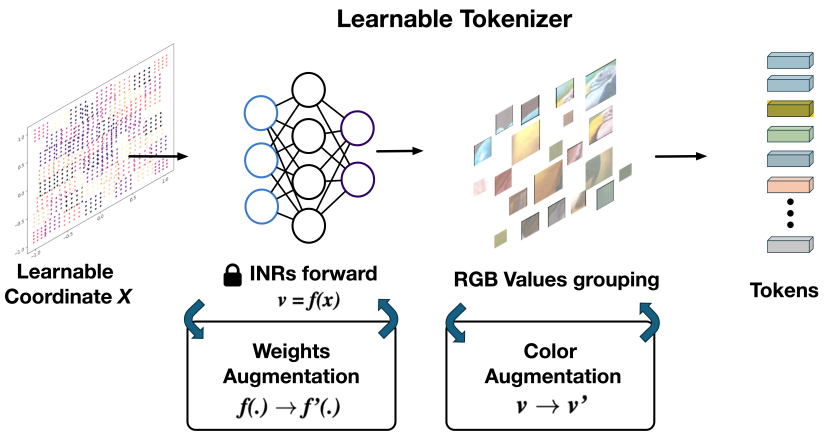


Figure 12: **Differential Augmentation** We propose geometric augmentation in weights-space and color augmentation in RGB-space.Following [72] we propose 15 differentiable transforms which enhance our dataset application.

Augmentation	Rotate	Translate	ShearX
Implementation	$W_t = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix}$	$b_t = W \triangle b$	$W_t = \begin{bmatrix} 1 & s \\ 0 & 1 \end{bmatrix}, b_t = W \begin{bmatrix} s \\ 0 \end{bmatrix}$

Table 6: **Implantation of geometric augmentation on weight-space**: We implement geometric transformation by modifying the weight W and bias b in first layer of INRs

can expressed as $x_{ij} = c_i + d_{ij}$. For (b) Learnable Scaling, we introduce a learnable scaling factor s_i for each patch. The queried coordinate is then given by $x'_{ij} = c_i + s_i d_{ij}$. For (c) Learnable Centers we make c_i themselves also learnable. Both method (b) and (c) keep the grid shape. For (d) Learnable pixels instead of learning a coordinate difference we directly make all coordinate $x_i j$ learnable. Finally in (d) we divide at beginning the coordinates randomly.Furthermore, to stabilize the training and ensure the learnable scale remains non-negative and the learnable pixels stay within the range $(-1, 1)$ a extra $Tanh(.)$ activation is applied on scaling factor and $Sin(.)$ is added on learnable coordaintes.

Differential augmentation As discussed in the main paper, differential augmentation is crucial to ensure that gradients can backpropagate to the learnable tokenizer. Unlike previous works [73, 74] hat train more INRs on augmented data, we implement [72] in a differential manner. Note that we propose to implement non-differential geometric augmentations in weight-space to make them differentiable. As shown in Fig 12, we first implement geometric augmentation such as Rotate, ShearX, ShearY, TranslateX, TranslateY, Cutout in by modifying the first layer of INRs. Specifically, we adjust the weights and biases as $W' = W + W_t$ and $b' = b + b_t$. This is calculated by $W'(x') + b' = Wx + b$ where x' is the corresponding coordinates after transformation.

Next, we implement color augmentations such as AutoContrast, Equalize, Solarize, Color Balance, Invert, Contrast, Brightness, and Sharpness in RGB space. Two main challenges arise: first, some color augmentations are non-differentiable, such as the Equalize operation, which creates a uniform distribution of grayscale values in the output image, and the Posterize operation, which reduces the number of bits for each color channel. To address that we first calculate the transform T outcome of these two operation and add the residual to our rgb value. $\triangle v = T(v) - v$ and $v' = v + \triangle v$. Note that we do not apply this residual addition to all color transformations because we want proposed learnable tokenizer to remain learn from color augmentations. Secondly some out-of-range RGB values can appear due to geometric augmentations $x' \notin (-1, 1)$, as illustrated in the Fig 13. These values can significantly impact downstream tasks such as segmentation and can also interfere with RGB augmentations process. To address this, we propose applying the same geometric transformations to a binary mask M with zero padding for out-of-range values. Before performing Color augmentations, we first apply the mask operation $v_{mask} = Mv$. We adopt some operations from [75].

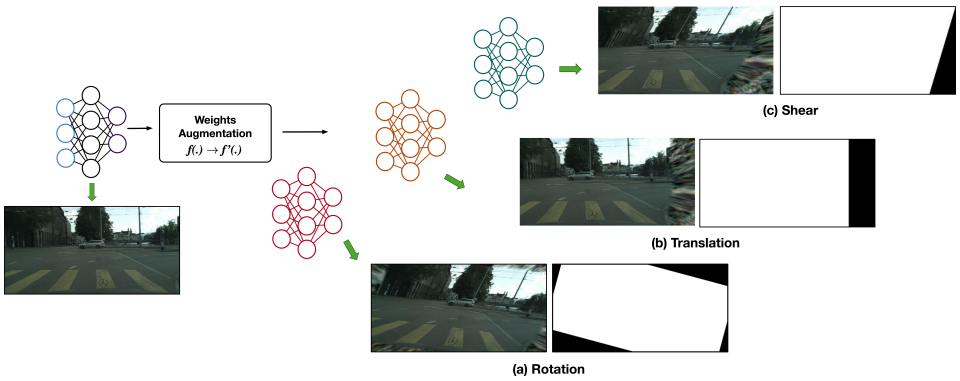


Figure 13: **Masking out-of-range values**: We implement zero-padding by applying same geometric transformation for a binary mask and we show examples for (a)rotation, (b)translation and (c)shear operation. Out-of-range RGB values are clearly visible in the bottom right corner of the transformed image.

3D implementation To lift up the learnable tokenizer to 3D volume tokenization we make following modification. Firstly we uniformly divide the space to N volumes and each include P^3 3D sample points. Then similarly for Learnable Centers + Learnable Scale we calculate the center points c_i for each volume $i \in \{1, 2, ...N\}$. The remaining operations are similar to 2D process.

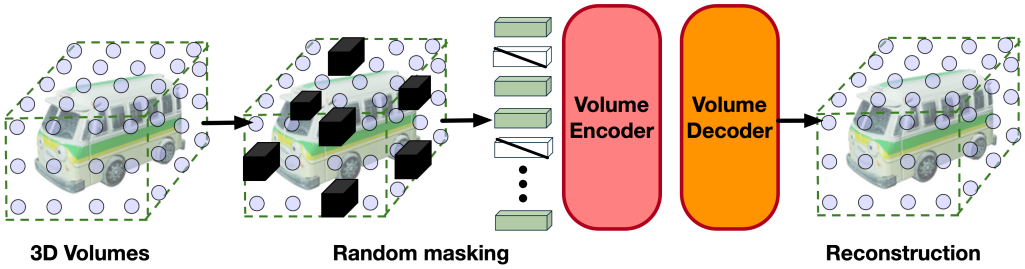


Figure 14: **Illustration of Proposed 3D Volume Encoder pretraining mechanism** We randomly mask out 80% of the volume tokens, allowing the encoder to operate only on the visible tokens. A small decoder processes the full set of encoded visible patches and masked tokens to reconstruct the input color volume and density volume. For qualitative results, please refer to Fig 15

7.3 Additional details of Benchmark

We provide detailed information on the implementation of our experiments and benchmarks. Additionally, we introduce our proposed pretraining mechanism, which has proven to be effective, as demonstrated in the main paper.

7.3.1 Implementation details

CIFAR-10-INRs We use ViT-tiny [76] with a depth of 12 and 3 heads for multi-head self-attention. The patch size is set to 4, and we employ a convolution layer with a kernel size of 4 as the embedding layer. For optimization we use Adam optimizer [77] with learning rate is 0.0001 for classify model and learnable tokenizer. Additionally, we apply CosineAnnealing leraning rate scheduler [78]. For regularization,we use regularize weight of $w_{reg} = 1$.

ImageNet-100-INRs: We utilize ViT-base [18, 76] with pretrained model 21k-1k and fine-tuned on our ImageNet-100 dataset.Our experiments are conducted with a batch size of 32, using distributed training on GeForce RTX 4090 GPUs. We employ the AdamW optimizer [79] with a learning rate of 1e-5 for the main model and the learnable tokenizer. A WarmupCosineAnnealingLR scheduler with one warmup epoch is used for learning rate adjustment.

Cityscapes-INRs: Similar to above we use AdamW optimizer [79] with learning rate 1e-4 for segmentation model and 1e-5 for learnable token. This is because in segmentation task we do not want misalignment between supervision area and tokenizaiton area too large. Following [19] we use PolynomialLR scheduler with power of 1.0.

Omniobject3D-INRs: We first introduce the pretraining mechanism, by following [67] we random masking the volume tokens as shown in Fig 14. Unlike [16] we follow the standard ViT structure and focus on pose regression task. The volume encoder operates only on the unmasked tokens. The proposed volume encoder operates only on the unmasked tokens and consists of 12 layers, each with 3 heads and an embedding feature dimension of 192. For the decoding process, we utilize 8 layers transformer-based decoder with same head numbers and embedding dimension as encoder. We use a shared and learnable masked token to fill in the originally masked-out positions and apply the positional encoding of the original tokens. The decoder then predicts the original RGB and density values, using mean squared error (MSE) as the loss function.

Next, we use the pretrained encoder for INRs pose regression, demonstrating its effectiveness with improved results across all proposed learnable tokenizers. We train the model for 100 epochs with a batch size of 8, where each batch includes 24 sampled views of a given scene. For non-pretrained experiments, we use a learning rate of 1e-4, while for the pretrained volume encoder, we use a learning rate of 1e-5.

7.3.2 Additional experiments

We conducted weight-space-only experiments and additional experiments on refined CIFAR-10-INRs, training for 500 epochs. We selected DWSNet [11] and HyperRepresentation [80] for benchmarking, as they represent two primary approaches: one proposes a permutation-invariant network structure to process trained INRs, while the other learn strong encoder to tokenizes the network weights to latent feature.

For DWSNet, we used a 4-layer model with a hidden dimension of 64. For HyperRepresentation, we employed an 8-layer transformer with a hidden dimension of 512. Notably, DWSNet performed poorly on the CIFAR-10 experiments, likely due to the lack of additional information from the input coordinate domain for the RGB 3D higher dimension output [11]. HyperRepresentation performed better but still yielded unsatisfactory results compared to other RGB space-based methods.

Method	Acc ↑	Precision ↑	F1 ↑
ViT[18]	84.28±0.41%	84.17± 0.44 %	84.25± 0.42 %
ViT[18] + LC	<u>85.11± 0.33%</u>	<u>85.03± 0.38%</u>	<u>85.09± 0.34%</u>
ViT[18] + LP + Reg	**85.35± 0.35%**	**85.33± 0.37%**	**85.34 ± 0.35%**
DWSNet[11]	38.12±1.32%	36.33±1.54%	37.11 ±1.33%
Hyper [80]	63.14 ± 1.12%	61.22 ± 1.45%	62.45 ± 1.34%

Table 7: **Refined CIFAR-10-INRs Classification.** We conduct additional experiments using refined CIFAR with 500 epochs.We report results from the weight-space-only method and observe that a performance gap still exists between the weights-only method and the RGB-based image method.

7.4 Additional information for Checklist

Potential negative societal impacts While our work on proposing a large-scale INRs dataset for 2D and 3D tasks offers significant advancements in the field of implicit neural representations, it is important to consider potential negative societal impacts such like (1) Privacy Concerns: The proposed dataset use other popular public dataset and share the same risk for privacy violations of other dataset. (2) Our work limit only to natural images and more diverse modality should be considered. (3) Environmental Impact: Training large-scale INRs models requires significant computational resources, which can contribute to high energy consumption and increased carbon footprint. This environmental impact is a growing concern with the proliferation of large-scale AI models.

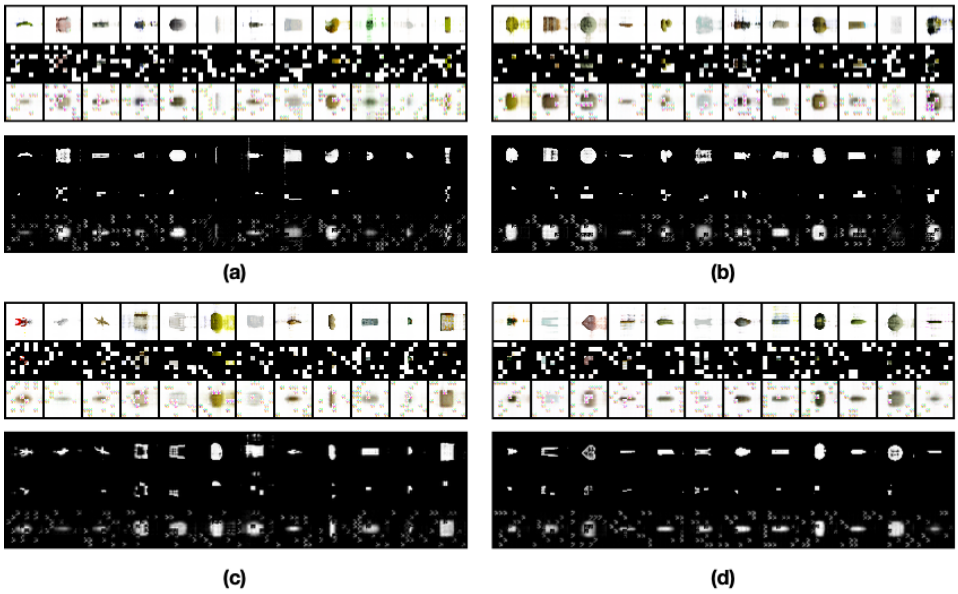


Figure 15: **Visualization on validation set of Omniobject3D reconstruction** We present four batches of reconstruction results in panels (a), (b), (c), and (d). The top row displays the input RGB volumes and density volumes. The middle row shows the masked volumes, while the bottom row illustrates our reconstruction results. Each sample contains 32x32x32 sampled points, and with a volume size of 4, it generates 512 tokens, of which only 102 are visible. Note that the output in known patches location may exhibit some artifacts.

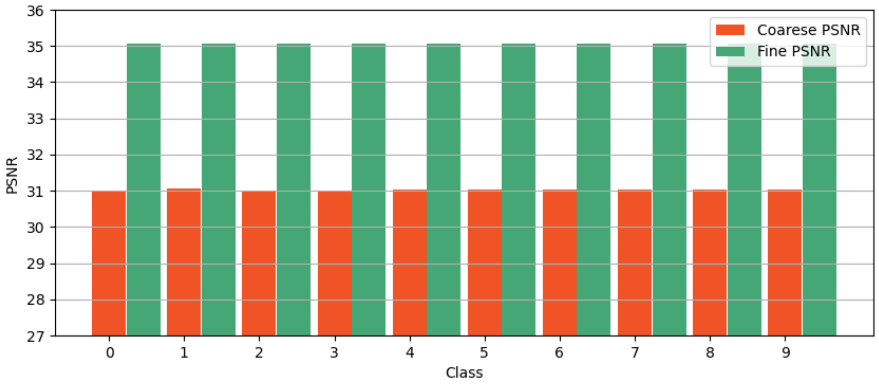


Figure 16: **PSNR for each class in CIFAR-10-INRs** We report the PSNR for all classes of CIFAR-10 INRs. Before refinement, the standard deviation across different tasks is 0.013, and after refinement, it is 0.005.

References

[1] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[3] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding, 2016.

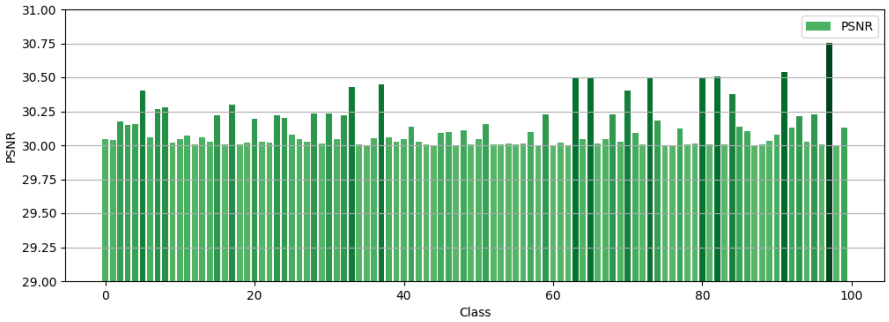


Figure 17: **PSNR for each class in ImageNet-100-INRs** We report PSNR on ImageNet-100 with standard deviation 0.157.

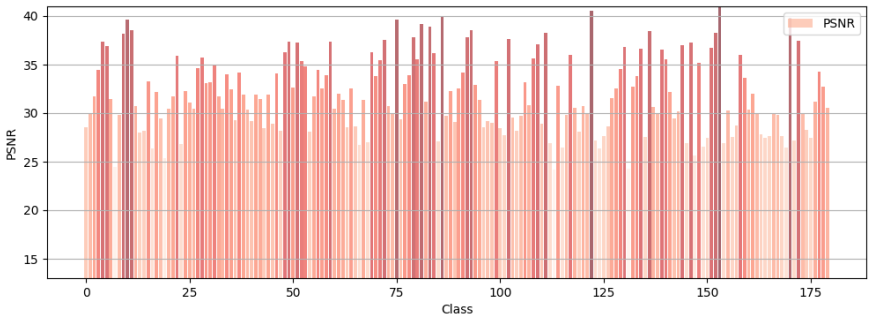


Figure 18: **PSNR for each class in Omniobject3D-INRs-INRs** We report PSNR on Omniobject3D with standard deviation 3.87.

[4] Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Jiawei Ren, Liang Pan, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, Dahua Lin, and Ziwei Liu. Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation, 2023.

[5] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020.

[6] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[7] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. In *Computer Graphics Forum*, volume 41, pages 641–676. Wiley Online Library, 2022.

[8] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019.

[9] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5939–5948, 2019.

[10] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019.

[11] Aviv Navon, Aviv Shamsian, Idan Achituve, Ethan Fetaya, Gal Chechik, and Haggai Maron. Equivariant architectures for learning in deep weight spaces, 2023.

[12] Luca De Luigi, Adriano Cardace, Riccardo Spezialetti, Pierluigi Zama Ramirez, Samuele Salti, and Luigi Di Stefano. Deep learning on implicit neural representations of shapes, 2023.

[13] Pierluigi Zama Ramirez, Luca De Luigi, Daniele Sirocchi, Adriano Cardace, Riccardo Spezialetti, Francesco Ballerini, Samuele Salti, and Luigi Di Stefano. Deep learning on 3d neural fields, 2023.

[14] Emilien Dupont, Hyunjik Kim, SM Eslami, Danilo Rezende, and Dan Rosenbaum. From data to functa: Your data point is a function and you can treat it like one. *arXiv preprint arXiv:2201.12204*, 2022.

[15] Matthias Bauer, Emilien Dupont, Andy Brock, Dan Rosenbaum, Jonathan Richard Schwarz, and Hyunjik Kim. Spatial functa: Scaling functa to imagenet classification and generation. *arXiv preprint arXiv:2302.03130*, 2023.

[16] Muhammad Zubair Irshad, Sergey Zakahrov, Vitor Guizilini, Adrien Gaidon, Zsolt Kira, and Rares Ambrus. Nerf-mae: Masked autoencoders for self-supervised 3d representation learning for neural radiance fields, 2024.

[17] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4):1–15, July 2022.

[18] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[19] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers, 2021.

[20] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5741–5751, 2021.

[21] Wenjing Bian, Zirui Wang, Kejie Li, Jia-Wang Bian, and Victor Adrian Prisacariu. Nope-nerf: Optimising neural radiance field with no pose prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4160–4169, 2023.

[22] Yue Chen, Xingyu Chen, Xuan Wang, Qi Zhang, Yu Guo, Ying Shan, and Fei Wang. Local-to-global registration for bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8264–8273, 2023.

[23] Dominic Maggio, Marcus Abate, Jingnan Shi, Courtney Mario, and Luca Carlone. Loc-nerf: Monte carlo localization using neural radiance fields, 2022.

[24] Lin Yen-Chen, Pete Florence, Jonathan T Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. inerf: Inverting neural radiance fields for pose estimation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1323–1330. IEEE, 2021.

[25] Arthur Moreau, Nathan Piasco, Dzmitry Tsishkou, Bogdan Stanciulescu, and Arnaud de La Fortelle. Lens: Localization enhanced by nerf synthesis. In *Conference on Robot Learning*, pages 1347–1356. PMLR, 2022.

[26] Sameera Ramasinghe and Simon Lucey. Beyond periodicity: Towards a unifying framework for activations in coordinate-mlps. In *European Conference on Computer Vision*, pages 142–158. Springer, 2022.

[27] Vishwanath Saragadam, Daniel LeJeune, Jasper Tan, Guha Balakrishnan, Ashok Veeraraghavan, and Richard G. Baraniuk. Wire: Wavelet implicit neural representations, 2023.

[28] Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function, 2021.

[29] Rohan Chabra, Jan E Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*, pages 608–625. Springer, 2020.

[30] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7154–7164, 2019.

[31] Shaohui Liu, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, and Zhaopeng Cui. Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2019–2028, 2020.

[32] Zhang Chen, Zhong Li, Liangchen Song, Lele Chen, Jingyi Yu, Junsong Yuan, and Yi Xu. Neurbf: A neural fields representation with adaptive radial basis functions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4182–4194, October 2023.

[33] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022.

[34] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. *ICCV*, 2021.

[35] Qi Ma, Danda Pani Paudel, Ajad Chhatkuli, and Luc Van Gool. Deformable neural radiance fields using rgb and event cameras, 2023.

[36] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. *ICCV*, 2021.

[37] Zhengqi Li, Qianqian Wang, Forrester Cole, Richard Tucker, and Noah Snavely. Dynibar: Neural dynamic image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[38] Alexandr Kuznetsov. Neumip: Multi-resolution neural materials. *ACM Transactions on Graphics (TOG)*, 40(4), 2021.

[39] Julian McGinnis, Suprosanna Shit, Hongwei Bran Li, Vasiliki Sideri-Lampretsa, Robert Graf, Maik Dannecker, Jiazhen Pan, Nil Stolt Ansó, Mark Mühlau, Jan S. Kirschke, Daniel Rueckert, and Benedikt Wiestler. Single-subject multi-contrast mri super-resolution via implicit neural representations, 2024.

[40] Junshen Xu, Daniel Moyer, Borjan Gagoski, Juan Eugenio Iglesias, P Ellen Grant, Polina Golland, and Elfar Adalsteinsson. Nesvor: Implicit neural representation for slice-to-volume reconstruction in mri. *IEEE Transactions on Medical Imaging*, 2023.

[41] Yu Sun, Jiaming Liu, Mingyang Xie, Brendt Wohlberg, and Ulugbek S Kamilov. Coil: Coordinate-based internal learning for imaging inverse problems. *arXiv preprint arXiv:2102.05181*, 2021.

[42] Qi Ma, Danda Pani Paudel, Ajad Chhatkuli, and Luc Van Gool. Continuous pose for monocular cameras in neural implicit representation, 2024.

[43] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 54(10s):1–41, 2022.

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[45] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

[46] Hugo Touvron, Matthieu Cord, and Hervé Jégou. Deit iii: Revenge of the vit. In *European conference on computer vision*, pages 516–533. Springer, 2022.

[47] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners, 2021.

[48] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers, 2021.

[49] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[50] Ali Hatamizadeh, Vishwesh Nath, Yucheng Tang, Dong Yang, Holger R Roth, and Daguang Xu. Swin unetr: Swin transformers for semantic segmentation of brain tumors in mri images. In *International MICCAI Brainlesion Workshop*, pages 272–284. Springer, 2021.

[51] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[52] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions, 2021.

[53] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers, 2020.

[54] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows, 2021.

[55] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers, 2021.

[56] Zilong Huang, Youcheng Ben, Guozhong Luo, Pei Cheng, Gang Yu, and Bin Fu. Shuffle transformer: Rethinking spatial shuffle for vision transformer, 2021.

[57] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal self-attention for local-global interactions in vision transformers, 2021.

[58] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvt v2: Improved baselines with pyramid vision transformer. *Computational Visual Media*, 8(3):415–424, 2022.

[59] Meng-Hao Guo, Cheng-Ze Lu, Qibin Hou, Zhengning Liu, Ming-Ming Cheng, and Shi-Min Hu. Segnext: Rethinking convolutional attention design for semantic segmentation. *Advances in Neural Information Processing Systems*, 35:1140–1156, 2022.

[60] Arash Amini, Arul Selvam Periyasamy, and Sven Behnke. Yolopose: Transformer-based multi-object 6d pose estimation using keypoint regression. In *International Conference on Intelligent Autonomous Systems*, pages 392–406. Springer, 2022.

[61] Arul Selvam Periyasamy, Arash Amini, Vladimir Tsaturyan, and Sven Behnke. Yolopose v2: Understanding and improving transformer-based 6d pose estimation. *Robotics and Autonomous Systems*, 168:104490, 2023.

[62] Arash Amini, Arul Selvam Periyasamy, and Sven Behnke. T6d-direct: Transformers for multi-object 6d pose direct regression, 2021.

[63] Thomas Georg Jantos, Mohamed Amin Hamdad, Wolfgang Granig, Stephan Weiss, and Jan Steinbrener. Poet: pose estimation transformer for single-view, multi-object 6d pose estimation. In *Conference on Robot Learning*, pages 1060–1070. PMLR, 2023.

[64] Zhendong Xiao, Changhao Chen, Shan Yang, and Wu Wei. Effloc: Lightweight vision transformer for efficient 6-dof camera relocalization. *arXiv preprint arXiv:2402.13537*, 2024.

[65] Xiaogang Song, Hongjuan Li, Li Liang, Weiwei Shi, Guo Xie, Xiaofeng Lu, and Xinhong Hei. Transbonet: Learning camera localization with transformer bottleneck and attention. *Pattern Recognition*, 146:109975, 2024.

[66] Jiankai Sun, Yan Xu, Mingyu Ding, Hongwei Yi, Chen Wang, Jingdong Wang, Liangjun Zhang, and Mac Schwager. Nerf-loc: Transformer-based object localization within neural radiance fields. *IEEE Robotics and Automation Letters*, 8(8):5244–5250, August 2023.

[67] Christoph Feichtenhofer, Yanghao Li, Kaiming He, et al. Masked autoencoders as spatiotemporal learners. *Advances in neural information processing systems*, 35:35946–35958, 2022.

[68] Shengju Qian, Yi Zhu, Wenbo Li, Mu Li, and Jiaya Jia. What makes for good tokenizers in vision transformer? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[69] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 776–794. Springer, 2020.

[70] Tomas Hodan, Daniel Barath, and Jiri Matas. Epos: Estimating 6d pose of objects with symmetries. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11703–11712, 2020.

[71] Ziya Erkoç, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. Hyperdiffusion: Generating implicit neural fields with weight-space diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14300–14310, 2023.

[72] Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space, 2019.

[73] Allan Zhou, Kaien Yang, Yiding Jiang, Kaylee Burns, Winnie Xu, Samuel Sokota, J Zico Kolter, and Chelsea Finn. Neural functional transformers. *Advances in Neural Information Processing Systems*, 36, 2024.

[74] Allan Zhou, Kaien Yang, Kaylee Burns, Adriano Cardace, Yiding Jiang, Samuel Sokota, J. Zico Kolter, and Chelsea Finn. Permutation equivariant neural functionals, 2023.

[75] E. Riba D. Mishkin D. Ponsa E. Rublee and G. Bradski. Kornia: an open source differentiable computer vision library for pytorch. In *Winter Conference on Applications of Computer Vision*, 2020.

[76] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface's transformers: State-of-the-art natural language processing, 2020.

[77] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[78] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2017.

[79] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

[80] Konstantin Schürholt, Dimche Kostadinov, and Damian Borth. Hyper-representations: Self-supervised representation learning on neural network weights for model characteristic prediction, 2022.