On the Few-Shot Generalization of Learning on Implicit Neural Representations

Vincent Tao Hu[1], David W. Zhang[1], Teng Long[1], Yunlu Chen[2],
Yuki M. Asano[1], Pascal Mettes[1], Efstratios Gavves[1], Basura Fernando[3], Cees G. M. Snoek[1]
[1]University of Amsterdam, [2]CMU, [3]CFAR, IHPC, A*STAR, Singapore

Abstract

Implicit Neural Representation (INR) is a promising representation pattern that can be used to unify different data types. It has been thoroughly explored in the areas of generation, discriminative modeling, and permutation symmetry. In this paper, we take a further step to explore the generalization ability of INR from the perspective of few-shot learning. Few-shot learning is a highly challenging task for implicit representations, as generalization needs to be performed from only a few labelled examples. We conduct in-depth analyses into what is needed to make INRs work for discriminative learning with limited samples, with the complexity and number of parameters of INRs, inherent data augmentation from INRs, the choice of activation function, and weight initialization as important factors towards generalization of this special representation. We hope that this direction will encourage further exploration by others.

1. Introduction

Implicit Neural Representations (INRs) [23, 19, 4, 33, 28] are coordinate-based neural networks that represent a field, such as an image, by mapping 2D coordinates to color in image space. The adoption of neural fields has garnered substantial attention in the realm of computer vision, serving as a versatile means to represent diverse signals encompassing images [14, 33], 3D shapes/scenes [23, 19, 20], video [2], audio [28], medical images [26], climate data [11], and signals on non-Euclidean domain [13].

While methods for simple object-level data explore embedding a collection of data signals into INRs with shared latent space [19, 4, 1, 3], more related works [28, 33, 20] focus on fitting one INR function per data rather than generalizing across a set of signals, which is empirically much easier to optimize for complex data with high fidelity, even for exceedingly intricate 3D shapes [6] or images. Notably, the construction of individual INRs for each distinct shape or image is not only viable but also considerably more feasible for real-world deployment. An explicit advantage lies in the fact that this strategy obviates the necessity of having access to the complete dataset to accurately fit an INR to a specific shape or image. The increasing traction garnered by such methodologies suggests that the practice of fitting an individual network per input sample is poised to evolve into a common practice within the realm of learning INRs.

Despite the advantages and increasing popularity, how to process and learn on INRs that encode signals in the function space is not as straightforward as conventional data representations. Moreover, modeling individual INRs with no generalization [5] brings up a more challenging task to learn on these INR networks, especially when generalizing from limited data is of concern. Recent works have exhaustively explored INRs with regard to their discriminative capabilities [21], symmetry design involving permutation [21, 36, 39, 38], and data compression [37, 10]. Nevertheless, the current state-of-the-art performance of CIFAR10 image INR classfication sits at 63.4% accuracy [39], significantly trailing the 99.5% accuracy achieved through traditional pixel-based methods [9]. This disparity underscores the challenges that underlie the classification of INRs.

This limiting capability of learning on INRs (especially for classification task) prompts a comprehensive exploration of the potential in the challenging landscape of few-shot learning [35, 29, 12], which is more demanding than general many-shot classification due to its need for strong generalization from few annotations. To the best of our knowledge, this work is the first attempt to subject INR-based classification to a rigorous stress-testing regimen of few-shot learning. By delving into this uncharted territory, we aim to gain insights into the adaptability and limitations of learning on INRs in scenarios with limited data, thereby extending the frontiers of their applicability. This research not only contributes to the evolution of INR-based learning but also serves as a guiding compass for future investigations of leveraging Implicit Neural Representations for high-level recognition beyond classification [39, 38, 21, 36], steering the trajectory of innovation in this dynamic field.

Our contributions are multifaceted. Firstly, this paper represents the inaugural foray into dissecting INRs through

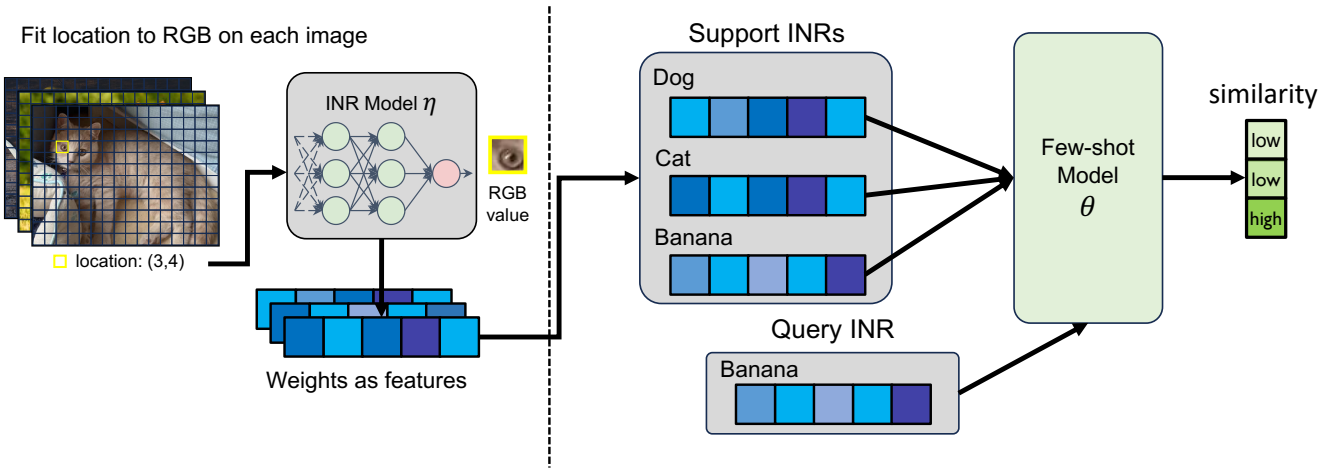

Figure 1. **Our method is composed of two decoupled parts.** 1). Addressing overfitting in the context of INRs by associating coordinates with RGB values, INR η is obtained in the end. 2). Utilizing the INRs η from the first step as the input representation for few-shot learning. We employ a similarity matching loss between support and query INRs to accomplish few-shot learning. The Few-shot model is parameterized by θ.

the prism of the few-shot framework. Secondly, we curate several datasets, ranging from easy to complex, to conduct an exhaustive dissection of various contributing factors. These factors include augmentation, activation functions, and initialization techniques, network architectures, and other pertinent factors. Lastly, we proffer a rudimentary yet competitive transformer-based baseline model. This baseline model serves as a crucial benchmark, intended to catalyze and propel advancements within this evolving domain.

2. Method

We outline our approach in two distinct segments. Initially, we extract the INR by correlating pixel coordinates with RGB values. Subsequently, in the second phase, we treat INR as the image's feature representation and employ it in the encoder $P(\cdot;\theta)$ as the input for the few-shot learning process. We will proceed to elucidate these steps individually.

2.1. Obtaining INR by overfitting

INRs are functions $\mathbf{x}_\eta : \mathcal{K} \rightarrow \mathcal{A}$ mapping *coordinates* $\mathbf{k} \in \mathcal{K}$ (e.g. pixel locations) to *features* $\mathbf{a} \in \mathcal{A}$ (e.g. RGB values) with parameters η. Given a data point as a collection of coordinates $\{\mathbf{k}_i\}_{i\in\mathcal{I}}$ and features $\{\mathbf{a}_i\}_{i\in\mathcal{I}}$ (where $\mathcal{I}$ is an index set corresponding to e.g. all pixel locations in an image), INRs are fitted by minimizing mean squared error over all coordinate locations:

$$\min_{\eta} \mathcal{L}(\mathbf{x}_\eta, \{\mathbf{k}_i, \mathbf{a}_i\}_{i\in\mathcal{I}}) = \min_{\eta} \sum_{i\in\mathcal{I}} \|\mathbf{x}_\eta(\mathbf{k}_i) - \mathbf{a}_i\|_2^2. \quad (1)$$

Hence each $\mathbf{x}_\eta$ is the Implicit Neural Representation (INR) corresponds to e.g. a single image. Typically, $\mathbf{x}_\eta$ is parameterized by a feedforward neural network (MLP) with positional encodings [20, 33] or sinusoidal activation functions [28] that allow fitting of high-frequency signals.

2.2. Few-Shot Generalization of Learning on INRs

We focus on few-shot classifier learning, a scenario involving three distinct datasets: meta-train, meta-val, and meta-test sets. In this setup, the support and query sets are characterized by a shared label space, while the meta-train set encompasses labels that remain distinct and do not intersect with those of the meta-val/meta-test sets. When addressing a specific few-shot challenge involving K labeled instances for each of C distinct classes present in the support set, we denote it as a C-way K-shot problem.

Relying solely on the support set, we are able to train a classifier that assigns class labels $\hat{\mathbf{y}}$ to samples $\hat{\mathbf{x}}$ within the query set. Yet, owing to the scarcity of labeled support samples, the performance of this classifier often falls short of expectations. To address this limitation, our goal is to engage in meta-learning using the training set. This endeavor seeks to distill transferable insights, thereby enhancing the efficacy of few-shot learning on the INR support set and leading to improved classification of the INR query set.

2.3. Input and Network Design

In the following section, we explore two structures for injecting inductive bias into the framework of few-shot learning: MLP and Transformer [34], as shown in Figure 2. However, the representation of $\mathbf{x}$ is not a perfect tensor structure that can be used by the MLP and transformer, as it is represented by weight and bias. In the following, we will explain how to organize the weight and bias into a suitable format to fit them into the MLP or transformer.

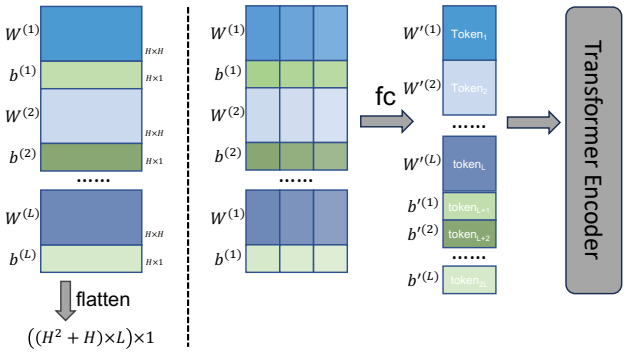

Figure 2. **Two different formulations of the weight-bias in INRs.** In the case of the left figure, we straightforwardly flatten the combination of weight and bias, creating a one-dimensional vector. Conversely, for the right figure, we input the merged weight-bias combination, treating it as an individual token within the transformer.

Vectorized weight and bias for MLP. We consider INRs composed of several hidden layers, each one with H neurons, *i.e.* the linear transformation between two consecutive layers is parameterized by a matrix of weights $\mathbf{W}_i \in \mathbb{R}^{H\times H}$ and a vector of biases $\mathbf{b}_i \in \mathbb{R}^{H\times 1}$. Thus, stacking $\mathbf{W}_i$ and ${\mathbf{b}_i}^T$, the mapping between two consecutive layers can be represented by a single matrix $\mathbf{M}_i \in \mathbb{R}^{(H+1)\times H}$. For an INR composed of $L+1$ hidden layers, we consider the L linear transformations between them. Hence, stacking all the L matrices $\mathbf{M}_i \in \mathbb{R}^{(H+1)\times H}, i = 1,\ldots,L$, between the hidden layers we obtain a single matrix $\mathbf{M} \in \mathbb{R}^{L(H+1)\times H}$ as shown in Figure 2. We use it to represent the INR x as the input of the encoder $P(\cdot;\theta)$. The encoder $P(\cdot;\theta)$ is typically formulated by a MLP.

Tokenize weight and bias for Transformer. We treat the weight and bias of each layer as a single token. For example, in a 2-layer network, we have a total of 4 tokens encompassing the weight and bias for both layers. Since the feature dimensions may differ, we also use a MLP to convert them into the same dimension. The converted tokens are concatenated and fed into a transformer network [34].

2.4. Training and Evaluation

Similarity Loss. We aim to establish a correspondence from a limited support set comprising k instances of image-label pairs denoted as $S = (\mathbf{x}_i, \mathbf{y}_i)_{i=1}^k$, leading to the creation of a classifier $c_S(\hat{\mathbf{x}})$.

We define the transformation $P(\hat{\mathbf{y}}|\hat{\mathbf{x}}, S;\theta) : S \rightarrow c_S(\hat{\mathbf{x}})$, with P being parameterized through a neural network. In detail, $P(\cdot;\theta)$ is formulated as:

$$P(\hat{\mathbf{y}}|S;\theta) = \frac{\sum_{i=1}^{k} e^{c(f(\hat{\mathbf{x}}),g(\mathbf{x}_i))}\mathbf{y}_i}{\sum_{j=1}^{k} e^{c(f(\hat{\mathbf{x}}),g(\mathbf{x}_j))}}, \tag{2}$$

where $S = (\mathbf{x}_i, \mathbf{y}_i)_{i=1}^k$ from the support set, and $c(\cdot,\cdot)$ is the cosine similarity metric, The functions f and g can act as feature extractors. Typically, in standard scenarios, the parameters of both f and g are shared.

Meta-training stage. Although, there exists several fine-tuning based meta-learning paradigm [12] in this regime. For simplicity, we opt for non fine-tuning based method e.g., [35, 29, 32] In general, the training loss can be formulated in the following:

$$\theta = \arg\max_{\theta} \mathbb{E}_{L\sim T}\left[\mathbb{E}_{S,Q\sim L}\left[\sum_{(\mathbf{x},\mathbf{y})\in Q} \log P\left(\mathbf{y}|\mathbf{x}, S;\theta\right)\right]\right]. \tag{3}$$

where S is the INR support set, Q denotes a batch of query data with INR $\mathbf{x}$ and corresponding label $\mathbf{y}$. A label set L sampled from a task T, $L \sim T$, will typically have 5 to 25 examples. T denotes the task set, which typically contains the full class set of training INRs.

Meta-evaluation stage. Evaluation using the following Equation results in a model that performs well when sampling $S' \sim T'$ from a different distribution of unseen labels.

$$\tilde{\mathbf{y}} = \arg\max_{\tilde{\mathbf{y}}} P\left(\tilde{\mathbf{y}}|\mathbf{x}, S';\theta\right). \tag{4}$$

This can serve as a simple indicator of the model's ability to generalize to unseen INRs.

Dataset normalization. We empirically found dataset normalization to be beneficial and aid training. The INRs are normalized as follows: let $\mathbf{x}_{[i]}$ denote the i-th layer of the INR in the dataset and let $\mathbf{x}_{[ij]}$ the j-th entry in $\mathbf{x}_{[i]}$, which can be two types, weight, and bias. Let $\mu(\mathbf{x}_{[ij]})$ denote the mean vector over the dataset and $\sigma(\mathbf{x}_{[ij]})$ the vector of standard deviations. We normalize each element as :

$$\mathbf{x}_{[kij]} \leftarrow (\mathbf{x}_{[kij]} - \mu(\mathbf{x}_{[ij]}))/\sigma(\mathbf{x}_{[ij]}), \tag{5}$$

where $\mathbf{x}_{[kij]}$ denotes the partial INR from i-th layer and weight/bias from k-th sample. Therefore, for every weight and bias from different layers, they approximately follow a normal distribution.

3. Experiments

In this section, we first introduce the details of the training and evaluation, followed by the details of our three INR datasets. Based on the organized dataset, we conduct extensive ablation studies across INRs complexity, weight initialization, normalization, and INRs augmentation. With these experiments, we shine light on the components that are key towards better discriminative image-based learning in INRs.

3.1. Training Details

For INRs, by default we use the SIREN [28] architecture, i.e., MLP with sine activation. We train the INRs using

Dataset	INR-Omniglot	INR-Fewshot-CIFAR100	INR-*mini*-ImageNet
train/val/test set	33/5/12[1]	60/20/20	64/16/20
Dataset Size	32,460	60,000	60,000
Image Resolution	32×32×1	$32 \times 32 \times 3$	84×84 × 3
Activation Function	SIREN	SIREN	SIREN
INR Structure: Low-complexity	2/32/32/1	2/64/64/3	—
INR Structure: High-complexity	2/64/64/64/64/1	2/64/64/64/64/64/3	2/256/256/256/256/3

Table 1. **Summary of three INR datasets** for exploring the generalization ability of INRs from few-shot learning.

Adam [16] for 1,000 steps with learning-rate $5e-4$. When the PSNR of the reconstructed image from the learned INRs is greater than 40, we use early stopping to reduce the generation time.

The transformer has a hidden size of 64 (i.e., the size of each token after linear projection), 4 layers, and 4 self-attention heads. For the training of few-shot learning, we adopt the Adam optimizer with a learning rate $1e-5$. We calculate the normalization mean and variance layer by layer on meta-train set. For the evaluation of the few-shot learning, following [29, 8], All methods are evaluated over 1,000 few-shot episodes unless noted otherwise. For implementation, we opt for the PyTorch [24] and Pytorch-Meta package.[2].

3.2. Dataset Details

In our study, we examine three datasets: INR-Omniglot [18], INR-Fewshot-CIFAR100, and INR-*mini*-ImageNet, which range in complexity from simple to challenging. We provide detailed descriptions of each of these datasets in the subsequent sections.

INR-Omniglot INR-Omniglot [18] is the derivative dataset of Omniglot, consisting of 1,623 characters from 50 different alphabets. Each of these was hand drawn by 20 different people. The large number of classes (characters) with relatively few data per class (20), makes this an ideal data set for testing few-shot learning on INRs. We exactly follow the dataset split from [18], and extract the corresponding INRs. Due to its simplicity and representativeness, all ablation studies in our experiment are carried out using the INR-Omniglot dataset.

INR-Fewshot-CIFAR100 We curate a derivative dataset for the purpose of few-shot learning on INRs, derived from the Fewshot-CIFAR100 dataset [22, 17]. We split the original dataset by superclass to minimize information overlap and make it more challenging than other datasets, such as INR-Omniglot. The original CIFAR100 dataset comprises 32×32 color images belonging to 100 different classes, with 600 images per class. These 100 classes are grouped into 20 superclasses. The train split contains 60 classes belonging to 12 superclasses, while the validation and test splits contain 20 classes belonging to 5 superclasses each. This split allows us to better understand the generalization ability across both classes and superclasses.

INR-*mini*-ImageNet To further explore the generalization ability of INRs on a more challenging dataset, we curate a dataset from the *mini*-ImageNet dataset, proposed by [35], contains 100 classes, each with 600 84×84 images. To perform meta-validation and meta-test on unseen INR tasks (and classes), we use 16 and 20 classes respectively, isolated from the original set of 100, leaving 64 classes for training tasks. We follow the same train/validation/test split as suggested by [25, 35].

3.3. Experimental findings

Below, we provide a series of ablation studies across different optimization axes that are important to perform discriminative learning for INRs in the challenging few-shot domain. The aim is to present a series of foundations to build upon for the long-term goal of obtaining high representational power for image-based INRs.

3.3.1 Higher complexity, higher accuracy

We present the main results for INR-Omniglot and INR-Fewshot-CIFAR100 in Tables 2 and 3, respectively. We compare primarily three methods:1). Zero-learning few-shot evaluation, where we do not train any network but instead compare the INR representations directly using similarity matching based on cosine distance. 2). An MLP is used as the encoder $P(\cdot;\theta)$ for conducting training, followed by the matching loss from MatchingNet [35]. 3). Transformer-based methods, where we first organize weights and biases into tokens and feed them into transformers.

We investigate both low- and high-complexity INR structures, with details in Table 1. Our network is trained using the 5-way 5-shot configuration. For this experiment, we utilize the most complete version of our method, with data augmentation from INRs, normalization, fixed weights, and sine-based activation functions. In later ablation studies, we will individually investigate all these choices with their impact on few-shot learning using INRs.

[2]https://github.com/tristandeleu/pytorch-meta

	Methods	5way - 1shot	5way - 5shot	10way - 1shot	10way - 5shot
Low-complexity INR	zero-learning	28.1± 0.14	30.1± 0.94	13.1± 0.51	14.3± 1.23
	MLP	29.1± 0.21	33.2± 0.33	14.2 ± 0.89	16.3± 0.53
	Transformer	38.91± 0.64	45.6± 0.93	23.82± 1.13	30.46± 0.94
High-complexity INR	MLP	33.1± 0.79	35.4± 0.63	17.2± 0.73	18.2± 0.68
	Transformer	56.2± 0.98	62.86± 0.94	40.14 ± 0.84	46.1 ± 0.98

Table 2. **Few-shot classification accuracies of INR-Omniglot.** We trained the network using the 5-way 5-shot configuration and evaluated it under different combinations of ways and shots. For the MLP approach, we adjust the number of layers and hidden units to align them appropriately with the parameter count of the corresponding Transformer.

	Methods	5way - 1shot	5way - 5shot	10way - 1shot	10way - 5shot
Low-complexity INR	zero-learning	22.1± 0.54	23.1± 0.45	11.2± 0.56	12.1± 1.32
	MLP	23.2± 1.32	24.2± 0.65	15.6 ± 1.23	13.4± 1.32
	Transformer	25.2± 0.68	27.4± 0.56	14.2± 1.4	16.34± 0.93
High-complexity INR	MLP	28.1 ± 0.78	30.5± 1.24	17.4± 0.74	20.3± 0.87
	Transformer	31.2± 0.89	36.9 ± 0.74	19.2 ± 0.39	22.58 ± 0.97

Table 3. **Few-shot classification accuracies of INR-Fewshot-CIFAR100.** We trained the network using the 5-way 5-shot configuration and evaluated it under different combinations of ways and shots. For the MLP approach, we adjust the number of layers and hidden units to align them appropriately with the parameter count of the corresponding Transformer.

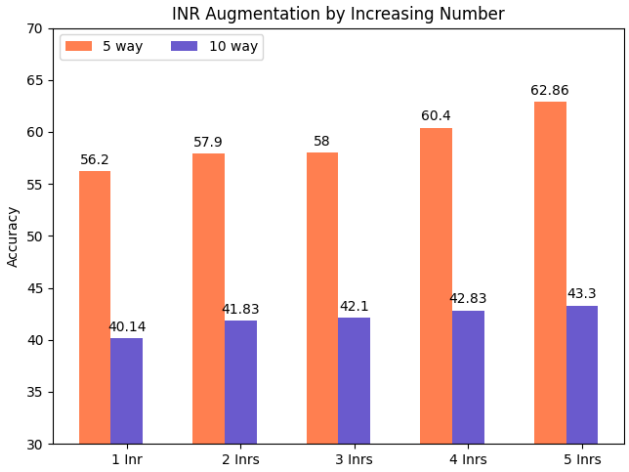

Figure 3. **Increasing INR augmentation view number from 1 to 5 can boost the performance in different way numbers.**

First and foremost, our observation indicates that INRs with higher complexity, characterized by the parameter η, can lead to substantial improvements when compared to low-complexity INRs. This conclusion holds true across various datasets, way numbers, and shot numbers. The "zero-learning" method signifies two key points. Firstly, the INR weights already exhibit a notable degree of distinctiveness, and this distinctiveness has the potential to escalate as more intricate INRs are employed. Secondly, the incorporation of MLP or Transformer components yields significant advantages when compared to the "zero-learning" approach. This observation underscores the necessity of introducing supplementary encoding to the initial INRs. This additional encoding plays a crucial role in facilitating the generalization of classification to INRs that were previously unseen.

Secondly, We note that a significant disparity remains when compared to image-space meta-learning. For instance, the present state-of-the-art result on the image Omniglot dataset approaches 98.9% in the 5-way 5-shot configuration [35]. This discrepancy highlights the long road ahead for the problem of INRs generalization. Importantly, our focus isn't solely on exploring INRs as a distinctive feature representation, but rather on scrutinizing its ability to generalize effectively to INR classes that haven't been encountered previously in training.

To the end, in Table 5, we further reveal that the complexity of the INR network plays a pivotal role in its performance across various shot numbers. Particularly in one-shot scenarios, a noticeable improvement of approximately 40% can be observed when compared to the performance of low-complexity INRs.

3.3.2 Use INRs as natural data augmenters

INRs allow us to enhance the limited data by incorporating multiple perspectives of INR for each image. This distinct INR viewpoint for each image is realized by employing slightly varied iteration numbers, such as 800, 900, 1000, 1100, 1200. As depicted in Figure 3, we consistently observe improvements as the count of augmented perspectives grows. From an intuitive standpoint, given that diverse perspectives are integrated into a single image, they can be perceived as distinct features of that image, thereby potentially boosting the classification accuracy. This is a unique feature of INRs that provide additional data complexity out-of-the-

box.

We also systematically increase the shot number across various way numbers, such as 5 and 10. Notably, our proposed approach consistently attains improvements through the utilization of multiple INR shots within the meta-learning context. This trend is evident in Figure 4. The trend underscores that distinct INRs originating from the same class indeed exhibit comparable encodings within our network. Consequently, they can collectively enhance overall performance.

3.3.3 Fixed weights benefit INRs

Because of its simplicity, we primarily explore two possible options: random initialization and fixed weight initialization. For fixed weight initialization, we optimize a random image to acquire a fixed weight, which then serves as the initialization for all other images within the INR extraction process. Derived from the data in Table 9, we can draw the conclusion that maintaining fixed weights yields superior performance when contrasted with using random weights. Our hypothesis is that the fixed weight initialization results in a more initialization-coherent INR embedding for meta-evaluation purposes.

3.3.4 Sine activations instead of ReLU

In Implicit Neural Representation, ReLU and SINE are the most common choice of activation functions. In Table 7, we compare ReLU to the sine-based activation functions in SIREN, under the sample network complexity, which is $2 \times 64 \times 64 \times 64 \times 64 \times 1$. We observe that SIREN can achieve a higher degree of generalization gain compared to ReLU. We therefore recommend this activation function over the standard choice in the field.

3.3.5 Don't forget normalization

In Table 8, we observe that, similar to discriminative methods in the image space [27, 7], normalization is necessary in INRs as well. We hypothesize that this is attributed to the normalization of the encoder's weights, which contributes to the optimization process.

The complexity of the encoder $P(\cdot;\theta)$ Our observation reveals that the complexity of the encoder, denoted as $P(\cdot;\theta)$, should not be excessively high to avoid the risk of overfitting in few-shot learning. Interestingly, this finding contrasts with the influence of the INR parameter η. We hypothesize that this discrepancy is attributable to the redundancy present in the INR representation, which places a constraint on how complex the encoder can be without becoming overly intricate.

Scaling up to more advanced datasets We have extended our investigation to assess performance on a more complex dataset, INR-*mini*-ImageNet, as shown in Table 4. The results are not as favorable as those achieved with INR-Omniglot and INR-Fewshot-CIFAR100, indicating the challenging nature of this particular task. The complexity of the dataset itself may contribute to this outcome. Additionally, the images in this dataset have a significantly higher resolution than the 32×32 resolution of the other datasets. We expect that the outcomes on this complex dataset will inspire further exploration, especially within the context of more advanced datasets.

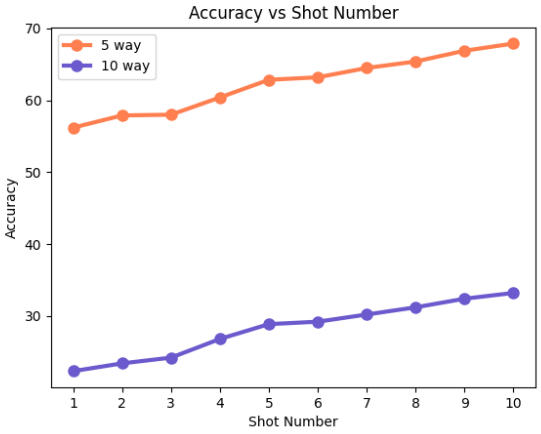

Figure 4. **Shot number *v.s.* accuracy.** Increasing the number of shots in meta-evaluation can help the network accumulate more accurate features from different prototypes.

4. Conclusion

This paper explores the generalization of Implicit Neural Representations (INRs) in the context of few-shot learning, considering their potential to offer consistent representations across diverse data types [2, 28, 26, 11]. To achieve this, we introduce a baseline founded on the transformer architecture. Additionally, we carefully analyze different factors, such as INRs complexity, model complexity, augmentation, weight initialization, and activation function of INRs. We hope that our exploration of INRs can inspire more interesting and broad applications of this special regime.

In future work, we are considering the improvement of how weights and biases are utilized by employing heuristic pruning techniques. Furthermore, there is a possibility to expand our investigation into domains like videos and point clouds, in order to provide additional support for the similar hypotheses proposed in this study. On a different note, we can also conduct a more thorough exploration of the impact of permutation symmetry [21, 36] within the weight-bias space, with the intention of incorporating a stronger inductive bias into the network's architecture. Lastly, we might

	Methods	5way - 1shot	5way - 5shot	10way - 1shot	10way - 5shot
High-complexity INR	zero-learning	22.1± 0.54	23.1± 0.45	11.2± 0.56	12.1± 1.22
	MLP	25.1 ± 0.54	26.5± 1.02	14.4± 1.24	15.3± 0.47
	Transformer	25.2± 0.94	27.9 ± 0.56	14.2 ± 0.95	16.58 ± 1.27

Table 4. **Few-shot classification accuracies of INR-*mini*-ImageNet.** We trained the network using the 5-way 5-shot configuration and evaluated it under different combinations of ways and shots. For the MLP approach, we adjust the number of layers and hidden units to align them appropriately with the parameter count of the corresponding Transformer.

	5-way INR-Omniglot	
INR structure	1-shot	5-shot
2 layer 32 dim	38.91± 0.64	45.6± 0.93
4 layer 64 dim	56.2 ± 0.98	62.86 ± 0.94

Table 5. **High-complexity INR is preferred.**

	5-way INR-Omniglot	
INR layers	1-shot	5-shot
First 2 layers	20.34 ± 1.81	25.30 ± 1.75
First 3 layers	42.31 ± 0.89	46.69 ± 0.74
First 4 layers	53.2 ± 0.98	56.86 ± 0.94
All	56.2 ± 0.98	62.86 ± 0.94

Table 6. **As more INR weights are included, the performance are better.** All weights are necessary.

	5-way INR-Omniglot	
Activation Function	1-shot	5-shot
ReLU	54.2 ± 1.32	59.1 ± 1.65
SIREN [28]	56.2 ± 0.98	62.86 ± 0.94

Table 7. **SIREN activation works better than ReLU**.

	5-way INR-Omniglot	
Method	1-shot	5-shot
w/o normalization	32.67 ± 1.81	34.2 ± 1.75
w/ normalization	56.2 ± 0.98	62.86 ± 0.94

Table 8. **INR Normalization is necessary** on the few-shot learning.

	5-way INR-Omniglot	
Initialization	1-shot	5-shot
random weight	55.2 ± 0.89	61.86 ± 0.49
fix weight	56.2 ± 0.98	62.86 ± 0.94

Table 9. **Fix weight initialization during INR overfitting is better.**

explore the application of generative modeling approaches, such as the diffusion model [30, 15, 31], to enhance the training process and mitigate the issue of overfitting in the Implicit Neural Representations.

References

[1] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *CVPR*, 2021. 1

[2] Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser Nam Lim, and Abhinav Shrivastava. Nerv: Neural representations for videos. *NeurIPS*, 2021. 1, 6

[3] Yunlu Chen, Basura Fernando, Hakan Bilen, Thomas Mensink, and Efstratios Gavves. Neural feature matching in implicit 3d representations. In *ICML*, 2021. 1

[4] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019. 1

[5] Thomas Davies, Derek Nowrouzezahrai, and Alec Jacobson. On the effectiveness of weight-encoded neural implicit 3d shapes. *arXiv*, 2020. 1

[6] Luca De Luigi, Adriano Cardace, Riccardo Spezialetti, Pierluigi Zama Ramirez, Samuele Salti, and Luigi Di Stefano. Deep learning on implicit neural representations of shapes. *arXiv*, 2023. 1

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. 6

[8] Guneet S Dhillon, Pratik Chaudhari, Avinash Ravichandran, and Stefano Soatto. A baseline for few-shot image classification. In *ICLR*, 2020. 4

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 1

[10] Emilien Dupont, Adam Goliński, Milad Alizadeh, Yee Whye Teh, and Arnaud Doucet. Coin: Compression with implicit neural representations. *arXiv*, 2021. 1

[11] Emilien Dupont, Yee Whye Teh, and Arnaud Doucet. Generative models as distributions of functions. *arXiv*, 2021. 1, 6

[12] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017. 1, 3

[13] Daniele Grattarola and Pierre Vandergheynst. Generalised implicit neural representations. *NeurIPS*, 2022. 1

[14] David Ha. Generating Large Images from Latent Vectors, Apr. 2016. 1

[15] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 7

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *AISTATS*, 2016. 4

[17] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. In *arXiv*, 2009. 4

[18] Brenden Lake, Ruslan Salakhutdinov, Jason Gross, and Joshua Tenenbaum. One shot learning of simple visual concepts. In *Proceedings of the annual meeting of the cognitive science society*, number 33, 2011. 4

[19] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, 2019. 1

[20] 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. In *ECCV*, 2020. 1, 2

[21] Aviv Navon, Aviv Shamsian, Idan Achituve, Ethan Fetaya, Gal Chechik, and Haggai Maron. Equivariant architectures for learning in deep weight spaces. *arXiv*, 2023. 1, 6

[22] Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. *NeurIPS*, 2018. 4

[23] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 1

[24] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *arXiv*, 2017. 4

[25] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017. 4

[26] Liyue Shen, John Pauly, and Lei Xing. Nerp: implicit neural representation learning with prior embedding for sparsely sampled image reconstruction. *IEEE Transactions on Neural Networks and Learning Systems*, 2022. 1, 6

[27] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *NeurIPS*, 2014. 6

[28] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *NeurIPS*, 2020. 1, 2, 3, 6, 7

[29] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017. 1, 3, 4

[30] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015. 7

[31] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021. 7

[32] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018. 3

[33] Matthew Tancik, Pratul P Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *NeurIPS*, 2020. 1, 2

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2, 3

[35] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *NeurIPS*, 2016. 1, 3, 4, 5

[36] David W Zhang, Miltiadis Kofinas, Yan Zhang, Yunlu Chen, Gertjan J Burghouts, and Cees GM Snoek. Neural networks are graphs! graph neural networks for equivariant processing of neural networks. In *ICMLW*, 2023. 1, 6

[37] Yunfan Zhang, Ties van Rozendaal, Johann Brehmer, Markus Nagel, and Taco Cohen. Implicit neural video compression. *arXiv*, 2021. 1

[38] Allan Zhou, Kaien Yang, Kaylee Burns, Yiding Jiang, Samuel Sokota, J Zico Kolter, and Chelsea Finn. Permutation equivariant neural functionals. *arXiv*, 2023. 1

[39] Allan Zhou, Kaien Yang, Yiding Jiang, Kaylee Burns, Winnie Xu, Samuel Sokota, J Zico Kolter, and Chelsea Finn. Neural functional transformers. *arXiv*, 2023. 1