PARTITION MATTERS IN LEARNING AND LEARNING-TO-LEARN IMPLICIT NEURAL REPRESENTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

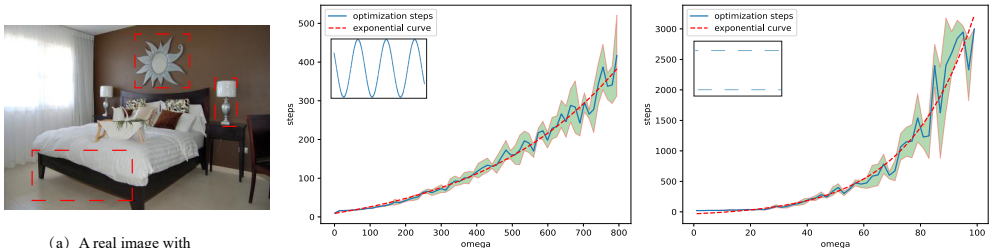
Implicit neural representation (INR) aims to learn a *continuous function* (i.e., a neural network) to represent an image, where the input and output of the function are pixel coordinator and RGB/Gray values, respectively. However, images tend to consist of many objects whose colors are not perfectly consistent, resulting in the challenge that image itself is actually a *discontinuous piecewise function* and cannot be well estimated by a continuous function. In this paper, we empirically investigate that if a neural network is enforced to fit a discontinuous piecewise function (e.g., a step function) to reach a fixed small error ϵ, the time costs will increase exponentially. We name this phenomenon as *exponential-increase* hypothesis. Obviously, handling an image with many objects is almost impossible in INR if this hypothesis is true. To address this issue, we first prove that partitioning a complex signal into several sub-regions and utilizing piecewise INRs to fit that signal can significantly reduce the convergence time, even when the exponential-increase hypothesis is true. Based on this fact, we introduce two partition-based INR methods: one for learning INRs, and the other for learning-to-learn INRs. Both methods are designed to partition an image into different sub-regions, and dedicate smaller networks for each part. In addition, we further propose two partition rules based on regular grids and semantic segmentation maps, respectively. Extensive experiments validate the effectiveness of the proposed partitioning methods in terms of learning INR for a single image (ordinary learning framework) and the learning-to-learn framework.

1 INTRODUCTION

Recently, an innovative model for data/signal representation called implicit neural representation (INR) has aroused researchers' great attention, due to their remarkable visual performance in computer vision tasks, including image generation (Sitzmann et al., 2020b), (Martel et al., 2021) and novel views synthesis (Mildenhall et al., 2021). To fit such an *implicit neural representation* for a 2D image, we usually learn a continuous function formalized by a neural network, which takes space coordinator $x \in \mathbb{R}^2$ as input and outputs the color values at the queried coordinate ($y \in \mathbb{R}^3$ if RGB and $y \in \mathbb{R}$ if gray).

However, we observe that images in the wild are usually very complex and consist of discrete objects with not perfectly consistent colors. A typical example is shown in Figure 1(a). This phenomena indicates that the in-the-wild images are actually *discontinuous piecewise functions*. There exist infinite/huge gradients on the boundaries between two discontinuous parts, which would prevent the neural network from converging to a small error (see red-rectangle areas in Figure 1(a)).

In this paper, we first empirically investigate that, the time complexity of fitting a discontinuous piecewise function with a single neural network would increase exponentially with respect to the number of discontinuous boundaries existing in the function space. For example, in Figure 1(b) and 1(c), we use a SIREN MLP (Sitzmann et al., 2020b) to fit a sinusoidal function $y = \sin(\omega x)$ and a step function $y = \text{sign}(\sin(\omega x))$, in which we can control the number of boundaries by choosing different ω. We then explore the relation between the required convergence step n and the frequency ω, and find that the relation curves align with the exponential function $n \propto O(p^n)$, where $p = 1.00248$ for sinusoidal function and $p = 1.0438$ for step function. We call this phenomenon



(a) A real image with discontinuous boundaries

(b) Fitting for sinusoidal functions

(c) Fitting for square wave functions

Figure 1: (a) Discontinuous parts exist obviously in regions with red boxes, which motivates us to use piecewise functions to represent the complex signals. (b) & (c) We dedicate a single neural network to fit $y = \sin(\omega x)$ and $y = \text{sign}(\sin(\omega x))$ and find that the number of convergence step is exponentially increasing with ω, which refers to the number of boundaries with high gradients in input domain.

as *exponential-increase* hypothesis. If this hypothesis is true, fitting an high-resolution in-the-wild image with a single continuous INR would become pretty hard.

Then, to address the issue caused by huge gradients in images, we mathematically prove that partitioning images into several parts and learning INRs within each part can significantly reduce the convergence time, even when the exponential-increase hypothesis is true. In light of this fact, we introduce partition-based INR methods and utilize partition in two INR frameworks: one for learning INRs, and the other for learning-to-learn INRs. Specifically, in both frameworks, we partition an image into different sub-regions based on particular rules, and dedicate smaller networks for each sub-region. We also propose two flexible partition rules: one is partitioning based on regular grids, and the other is partitioning based on semantic segmentation maps. Both of them can speed up the convergence of learning INRs as well as learning-to-learn INRs.

The extensive experiments show that our partition methods provides up-to 100% more efficient training process for single INR learning framework and provides more than 1.6 dB in average PSNR quality for learning-to-learn INR framework. The results verify the effectiveness of partitioning images on both learning INR for a single image (ordinary learning framework) and learning an algorithm that can learn good INRs (learning-to-learn framework).

2 Related Work

Implicit Neural Representations. Implicit Neural Representations (INR), also called coordinate-based neural networks (Xie et al., 2022), are emerging topics of interests in artificial intelligence community. By mapping a coordinate x to a quantity with a neural network (e.g., MLP), these continuous representations have shown great potentials in 3D scene reconstruction (Park et al., 2019), (Michalkiewicz et al., 2019), (Hao et al., 2020)), digital humans tasks (Yenamandra et al., 2021), (Saito et al., 2019), (Saito et al., 2021)), 2D images generation (Sitzmann et al., 2020b), (Skorokhodov et al., 2021), 3D shape and appearance generation (Mildenhall et al., 2021), (Schwarz et al., 2020), (Niemeyer & Geiger, 2021), physics-informed problems (Raissi et al., 2019), (Pfrommer et al., 2020) and so on. A lot of researches have been conducted on different aspects of implicit neural representations, such as the prior learning and conditioning (Sitzmann et al., 2019), the computation and memory efficiency (Meister et al., 2021), the expression capacity (Rebain et al., 2021), the edit ability (Sitzmann et al., 2019) and the generalization across different samples (Sitzmann et al., 2020a), (Tancik et al., 2021). For detailed review on implicit neural representations, we refer the readers to (Xie et al., 2022).

Partition Techniques. When scaling up to signals with large domain, INR always fail due to the high non-linearity of mapping function (Ren et al., 2013) as well as heavy inference time. Therefore, partition/decomposition techniques with discrete data structures are extensively employed. DeRF (Rebain et al., 2021) utilized Voronoi spatial decomposition on Neural Radiance Fields and obtained more efficient inference speed and better render performance than original NeRF (Mildenhall et al., 2021). KiloNeRF (Reiser et al., 2021) demonstrated that the rendering process of NeRF can be accelerated by three orders of magnitude by utilizing thousands of tiny MLPs, each of which only needs to represent parts of the scene. ACORN (Martel et al., 2021) used a multiscale block-

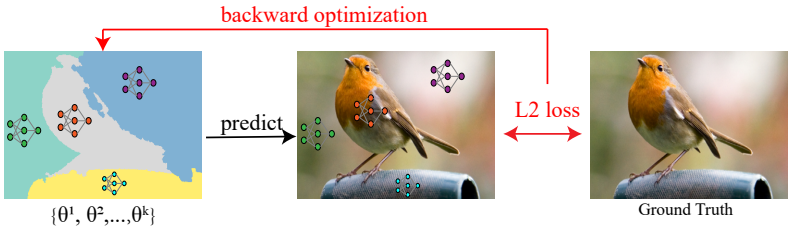


Figure 2: Framework for partition-based Learning INRs

coordinate decomposition to adaptively decompose the image domain, and successfully fitted gigapixel images with nearly 40 dB peak signal-to-noise ratio. Distinguished from above methods, the partition methods proposed in this paper focus on faster training of INRs rather than inference.

Learning-to-learn INRs. Meta-learning frameworks are always applied to train a meta-learner for INRs that can quickly adapt to new and unseen tasks with only few training examples. Sitzmann et al. (2020a) first introduced Model-Agnostic Meta-Learning(MAML) (Finn et al., 2017) into the field of INRs and their work learned excellent priors over the respective function space, leading to faster test-time inference and better geometry reconstruction from partial or noisy observations. Tancik et al. (2021) re-produced such findings for a wider variety of underlying signal types and introduced Reptile (Nichol et al., 2018) to reduce the memory cost of the inner loop on more complex signals. On top of their researches, we would show that partition methods can further improve the quality of learning-to-learn INRs framework when adapting to a new task.

3 Partition for Learning and Learning-to-Learn INRs

3.1 Motivations

Considering a field $\mathbf{q}$, INR learns a function Φ with parameters Θ to fit it, denoting such process as $\mathbf{q} = \Phi(\mathbf{x}; \Theta)$. SIREN (Sitzmann et al., 2020b) has shown that classical MLPs with ReLU activation fail to represent a signal's spatial and temporal derivatives, and proposed to leverage periodic activation functions to representing complex natural signals and their derivatives. However, even though SIRENs are able to represent complex natural signals, a lot of optimization steps are required due to the fact that too many boundaries with large gradients exist in complex signals domain. We argue that the events of successfully representing each boundary with large gradient by the neural network parameters Θ are independent with each other. Then we establish the following hypothesis:

Hypothesis 1. *Denote the complexity that one boundary with large derivatives is represented by Θ as p, then the complexity that all boundary with large derivatives are represented would be $O(p^n)$, where n is the number of boundaries with large derivatives within the domain.*

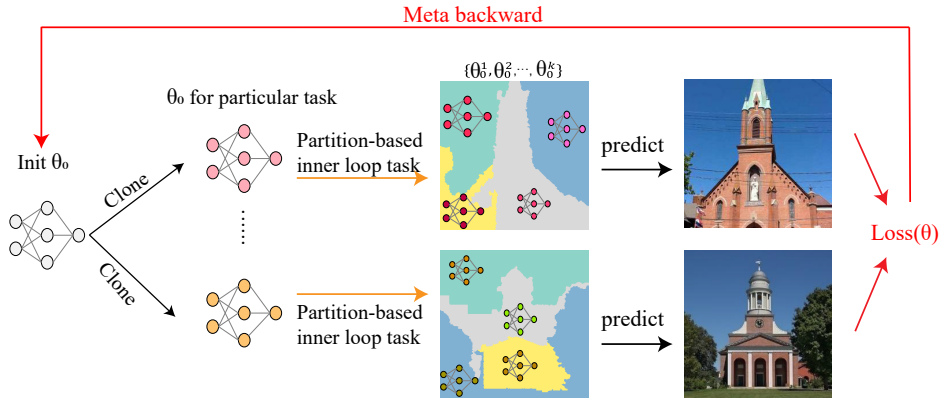
Experiments to demonstrate this hypothesis are shown in Figure 1. Then to reduce the high complexity of fitting n boundaries with a single MLP, we apply partition techniques on this problem. More specifically, we divide the whole input domain into several smaller sub-domains and leverage separate MLPs to fit a piecewise function to represent the whole function. We argue that the optimizations for all MLPs are parallel, thus we can establish the following proposition:

Proposition 1. *If the whole domain is divided into k sub-domains, the number of boundaries with large gradients falling in each sub-domain is $\{n_1, n_2, ..., n_k\}$, where $n = \sum_{i=1}^{k} n_i$. In case of $k \geq 3$, the complexity of dedicating neural networks with full capacity for each sub-domain is less than the complexity of representing the whole domain with a single neural network.*

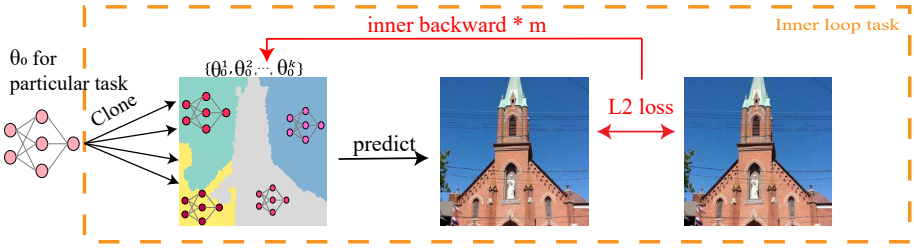
Proof. Since we dedicate neural networks with full capacity for each sub-domain, we can assume the complexity of fitting one boundary is still p, then the total complexity of parallelly fitting all sub-domains with separate neural networks is

$$p^{n_1} + p^{n_2} + ... + p^{n_k}. \tag{1}$$

Defining $\hat{n} = \max(n_1, n_2, ..., n_k)$, we hold Inequation 2:



(a) Framework for partition-based learning-to-learn INRs



(b) Inner loop task for learning-to-learn INRs

Figure 3: (a) Framework for partition-based learning-to-learn INRs. A meta-learner is applied to sample tasks of learning INRs and learns an initialized weight that can quickly fine-tune to a new image. (b) Partition-based inner loop task. The initialized weights would be copied and then optimized for each head within its corresponding sub-domain. (please view this figure in color version.)

$$\frac{p^{n_1}+p^{n_2}+...+p^{n_k}}{p^{\hat{n}}}=\frac{p^{n_1}}{p^{\hat{n}}}+\frac{p^{n_2}}{p^{\hat{n}}}+...+\frac{p^{n_k}}{p^{\hat{n}}}\leq 1+1+...+1=k. \tag{2}$$

Empirically, we should optimize each neural network for several times, so we have $p^{n_i}\geq 2$, then the following inequation holds:

$$\frac{p^{n}}{p^{\hat{n}}}=\frac{p^{(n_1+n_2+...+n_k)}}{p^{\hat{n}}}=\prod_{n_i\neq\hat{n}}p^{n_i}\geq 2^{k-1}. \tag{3}$$

In case of $k\geq 3$, we have $2^{k-1}>k$ and Proposition 1 is proved. □

Theoretically, with larger k, Proposition 1 can be generalized to the case of fitting each sub-domain with smaller neural networks, whose complexity of fitting one boundary is larger than p. We show the proof and the discussion of this case in Appendix A.

By now, we have mathematically proved that partition can speed-up the convergence of INRs. And we will present how we practically utilize the partition methods in INRs in the following sections.

3.2 Partition for Learning INRs

In this part, we show how we can apply partition techniques on learning INRs for 2D images. The framework of partition-based single INR method is shown in Figure 2. We propose to model a INR of a 2D image I as a weighted sum of k separate neural networks (denote as heads). Mathematically, this process can be written as

$$I(x)=\sum_{n=1}^{k}\omega_{\phi}^{n}(x)I_{\theta_n}(x), \tag{4}$$

where n denotes the head index, and $\omega_\phi^n(x) : \mathbb{R}^2 \mapsto \mathbb{R}^k$ is the partition mask that satisfies the property $\|w_\phi(\mathbf{x})\|_1 = 1$, meaning that each coordinate is only represented by one head.

In practice, we explore two different partition rules for 2D images, one is partition based on segmentation maps for 2D images and the other one is partition based on regular grids. Detailed discussion of these two partition rules is in Section 4.

3.3 Partition for Learning to Learn INRs

Sitzmann et al. (2020a) and Tancik et al. (2021) have shown meta-learning algorithms can provide excellent initial weight parameters for implicit neural representations, which leads to faster convergence and better generalization. In this part, we show that our partition methods can combine with meta-learning for INRs, and lead to better generalization and more flexible inference process than original meta-learning for INRs.

Considering a dataset of observations of signals T from a particular distribution $\mathcal{T}$ and a fixed number of optimization steps m, the meta-learning algorithms for INRs seek to find an initial weight θ_0^* that will result in the lowest possible final loss $L\left(\theta_m\right)$ if optimizing a network f_θ for m steps to represent a new signal from $\mathcal{T}$:

$$\theta_0^* = \arg\min_{\theta_0} E_{T\sim\mathcal{T}}\left[L\left(\theta_m\left(\theta_0, T\right)\right)\right]. \tag{5}$$

Combining with partition techniques, we partition the whole input domain into k sub-domains with partition rule ω, and seek to find an initial weight θ_0^* that serves as the initial weight of each head for each sub-domain and would result in the lowest possible final total loss when optimizing a set of network $F = \{f_{\theta^1}, f_{\theta^2}, ..., f_{\theta^n}, \}$, each of which will represent a part of the new signal from $\mathcal{T}$:

$$\theta_0^* = \arg\min_{\theta_0} E_{T\sim\mathcal{T}}\left[\sum_{n=1}^{k} L\left(\theta_m^n\left(\theta_0, T, \omega\right)\right)\right]. \tag{6}$$

We follow MAML (Finn et al., 2017) to learn such an initial weight which can serve as a good starting point for gradient descent for all heads. More specifically, given a task T and a number of optimizations steps m, our partition-based learning-to-learn INRs framework refers these task-specific optimization steps as inner loop, and wraps an outer loop to sample different signals T_j from $\mathcal{T}$ and generates their corresponding partition rules ω_j to learn the initial weight θ_0^*. Denote the parameters for the head k at the i step of inner loop and the j step of outer loop as $(\theta_i^k)_j$, the updated rule is defined as following:

$$(\theta_0)_{j+1} = (\theta_0)_j - \beta\nabla_\theta \sum_{n=1}^{k} L\left(\theta_m^n\left((\theta_0)_j, T_j, \omega_j\right)\right). \tag{7}$$

We conduct our experiments in 2D image reconstruction, and direct point-wise observations of the signal T are available. Therefore, we can supervise F with gradient descent using simple L2 loss:

$$L(\theta) = \sum_i \left\|F\left(\mathbf{x}_i\right) - T\left(\mathbf{x}_i\right)\right\|_2^2. \tag{8}$$

So far, we have presented our partition-based learning INRs method as well as partition-based learning-to-learn INRs method. The architectures of these two methods are presented in Figure 2 and 3 respectively.

4 Implementations

In this section, we introduce two partition rules that both work well under our frameworks. One is to partition based on regular grids (PoG for short) and the other is to partition based on semantic segmentation maps (PoS for short).

Partition based on regular grids. A simple but efficient partition rule is using regular grids to decompose the whole input domain. This method is widely used in image processing tasks based on ViT (Dosovitskiy et al., 2020), while Rebain et al. (2021) and Reiser et al. (2021) have discussed the

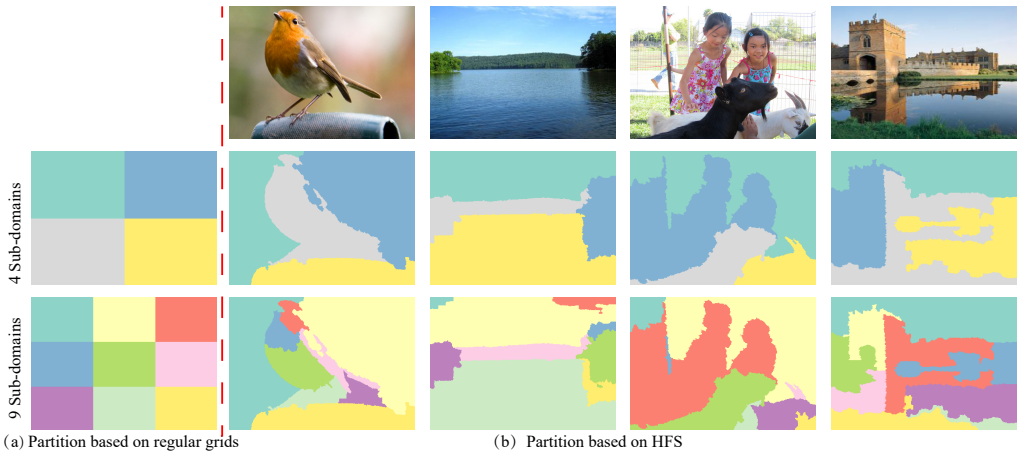


Figure 4: (a) Visualization of partition based on regular grids (PoG). (b) Visualization of partition based on HFS semantic segmentation maps (PoS). (please view this figure in color version.)

effect of regular grids decomposition in neural radiance fields tasks. Specifically, for 2D images, we subdivide the input domain into uniform grids of resolution $\mathbf{r} = (r_x, r_y)$, and utilize an independent neural network to fit the content within each grid. Therefore the mapping function m from pixel position $\mathbf{x}$ to its corresponding neural network index is defined as:

$$m(\mathbf{x}) = \left\lfloor \frac{\mathbf{x}}{\mathbf{r}} \right\rfloor. \tag{9}$$

Partition based on Semantic Segmentation Maps. We also seek a more flexible and reasonable partition rule, due to the fact that the real images contain non-homogeneous structures and the regular grids partition may violate the continuity of the images. Considering that a image always consists of several parts, it is reasonable to define a sub-domain as the region in which all pixels belong to the same part. Therefore, we seek a partition rule based on image semantic segmentation maps.

Specifically, we start with a hierarchical feature selection (HFS) (Cheng et al., 2016) algorithm, which is a rapid image segmentation system and reports over-segmentation results. The over-segmentation results usually assign the regions that are not connected with the same labels and the number of regions is always too much. Thus, we apply connected-components algorithm on the initial segmentation result to re-label those unconnected parts. And we finally apply a greedy region merging algorithm to obtain particular number of segmentation regions.

The performance of PoG and PoS methods are demonstrated in Figure 4 and the formalization of PoS algorithm is presented in Appendix B.

5 Experiments

In this part, we are going to demonstrate how our two partition rules can help to speed up the convergence of learning INRs as well as learning-to-learn INRs. We will first compare the convergence speed for learning the INR for a single in-the-wild image with different MLP architectures under the condition of taking partition or not taking partition. We would prove that our partition methods achieve better performance on two different INR architectures. We then choose SIREN (Sitzmann et al., 2020b) as our basic model and follow Sitzmann et al. (2020a)'s setting to train meta models with or without partition. The results demonstrate that partition-based learning-to-learn INRs method leads to higher quality with the same number of optimization steps for a new image.

5.1 Partition-based Learning INRs

Settings. We first choose a typical landscape image with dimension 380×254 and try to learn an implicit neural representation of this image. We choose two popular neural network architectures: one is SIREN MLP with periodic activation functions (Sitzmann et al., 2020b) and the other is MLP with classical ReLU activation functions with positional embedding. More specifically, we implement these two INR architectures as our baselines as following:

Table 1: Parameter table for all implemented models. All models contain 3 hidden layers.

Base architecture	#hidden layers/head	#head	#Total parameters
SIREN	512	1	790017
SIREN w. partition	256	4	793604
ReLU MLPs	512	1	911873
ReLU MLPs w. partition	240	4	926404

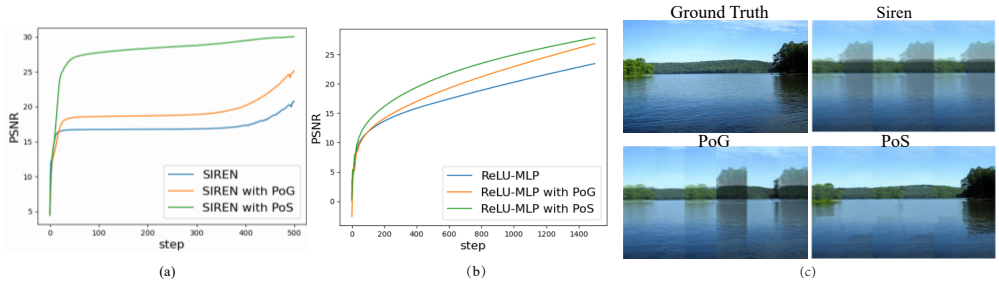


Figure 5: (a) PSNR vs. step curves for SIREN-based models. (b) PSNR vs. step curves for ReLU-MLP-based models. We can clearly observe that our partition-based models lead to faster convergence on both models. (c) Visual results of optimizing all models for 100 steps. The model with PoS method achieves best performance.

- SIREN: we use sinusoidal function as the activation function for each layer ($y = \sin(\omega x)$), setting up the ω for the first layer as 60 and the ω for the hidden layers as 30.
- ReLU-MLP with positional embedding: given an input x, we use a harmonic embedding layer to convert each feature in x into a series of harmonic features. We set up the number of harmonic functions as 60.

On top of these two baselines, we implement our two partition rules. To fairly demonstrate the effect of our partition methods, we dedicate neural networks with the same architecture and hyper-parameters but smaller capacity to fit each sub-region, namely heads. We guarantee that the total capacity of all heads is close to the capacity of baselines. In detail, we set up the number of hidden layer of all models as 3 and the capacity of all models are defined in Table 1. Following Sitzmann et al. (2020b)'s implementation, we apply Adam optimizer with a learning rate of $1e-4$. To make a comparison between two partition methods, we fix the number of partitioned regions to 4.

Results. We first show the PSNR curves with respect to optimization steps for applying partition methods on both two baselines, which are shown in Figure 5. We can see on both baseline architectures, our two partition rules lead to faster convergence, while partition based on semantic segmentation maps has better performance than partition based on regular grid. We can observe that for SIREN-based architecture, the model with PoS would converge to a high PSNR with very limited steps (less than 100), while the original SIREN needs more than 500 steps to converge to the same PSNR value. For ReLU-MLP-based architecture, the required steps of three cases that the PSNR value reaches 20 are 957, 672, 445 respectively, which indicates that our partition method based on regular grid (PoG) would lead to 50% speed-up while the partition method based on segmentation maps (PoS) leads to 100% speed-up.

We then take a closer look to the training process of the models developed from SIREN (Sitzmann et al., 2020b) and explore why the classical SIREN model fails when fitting a large scale image. As shown in Figure 5(c), we find that the classical SIREN model tends to generate artifacts due to the periodicity. And our partition methods reduce the space size that one single SIREN MLP need to represent, which partly alleviates the negative effect of the periodicity of the SIREN.

Robustness. To prove the robustness of our partition methods, we also evaluate our methods on a collection of in-the-wild images. More specifically, we collect 20 images with bedroom scenes from LSUN image dataset (Yu et al., 2015). We explore the mean PSNR values with 300 optimization steps for SIREN-based models and 1000 optimization steps for ReLU-MLP-based models. The results are reported in Table 2. We can observe that both of our partition methods drive the models to a higher PSNR value with the same optimization steps.

Table 2: Mean PSNR values for 20 test images. We optimize the SIREN-based models for 300 steps and ReLU-MLP-based models for 1000 steps.

Methods	PSNRs(mean $\pm$ std)$\uparrow$	Methods	PSNRs(mean$\pm$ std)$\uparrow$
SIREN	21.211 $\pm$ 7.089	ReLU-MLP	19.844 $\pm$ 1.835
SIREN w. PoG	23.864 $\pm$ 6.047	ReLU-MLP w. PoG	22.672 $\pm$ 2.848
SIREN w. PoS	**24.485** $\pm$ **4.450**	ReLU-MLP w. PoS	**22.863** $\pm$ **2.111**

Table 3: Mean PSNR values performance for SIREN models with different training and fine-tuning methods. For short, we rewrite G as the PoG partition method and S as the PoS partition method. We mark a superscript G/S if we apply corresponding method in training stage and mark a subscript G/S if we apply corresponding method in fine-tuning stage.

Setting	PSNR $\uparrow$		Setting	PSNR $\uparrow$	
	1 View	3 View		1 View	3 View
SIREN	19.42	22.60	-	-	-
SIREN_{G}	19.64	22.95	SIREN_{S}	19.85	23.09
$\text{SIREN}_{\text{G}}^{\text{G}}$	19.77	**24.23**	$\text{SIREN}_{\text{S}}^{\text{S}}$	**20.00**	23.93
$\text{SIREN}_{\text{G}}^{\text{S}}$	18.11	20.33	$\text{SIREN}_{\text{S}}^{\text{G}}$	19.93	23.89

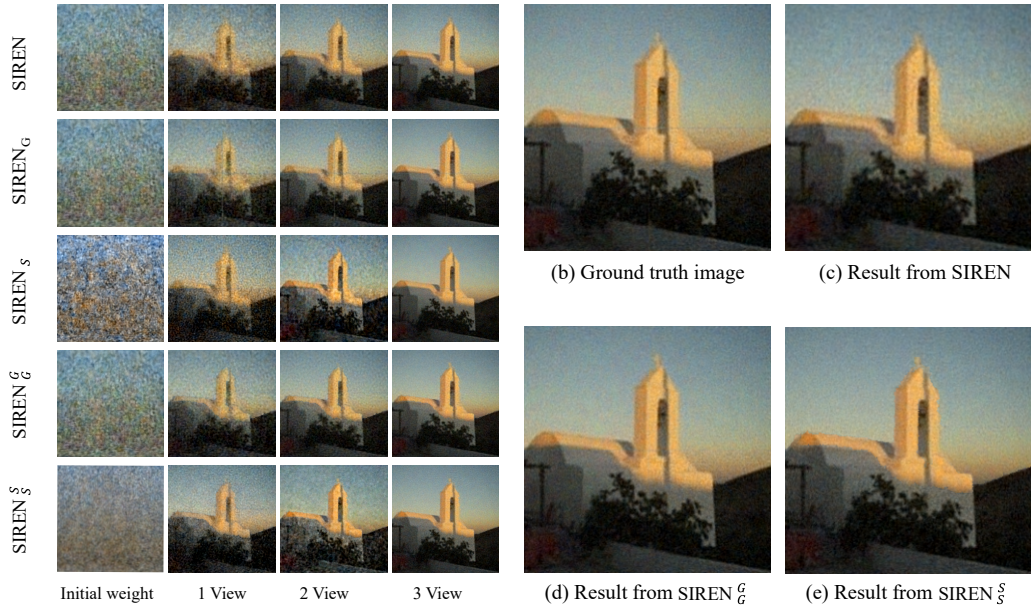
5.2 Partition-based Learning-to-learn INRs

Settings. To verify the effect of our partition-based Learning-to-learn INRs framework, we follow Sitzmann et al. (2020a) and apply our partition methods in MAML framework to learn an initial weight that can quickly fine-tune to an unseen image. We choose the outdoor church images from LSUN dataset (Yu et al., 2015) with the size of 256×256. The training set contains about 126k images and the test set contains 300 images. Following Sitzmann et al. (2020a) and Tancik et al. (2021), we choose SIREN as our basic model and set up the number of inner loop step N as 3, which means that our model would see each image for only three times. We apply per-parameter-per-step inner learning rate strategy with initial learning rate $\alpha = 1e - 5$. We train all of the meta-models with outer loop learning rate $\beta = 1e - 4$ and a batch size of 4.

Since our partition methods would duplicate the initial weight for each head, we maintain one copy of per-parameter-per-step learning rate for a single head but shared by all heads. We implement the baseline SIREN model that contains 3 hidden layers and 128 hidden features of each layer, which is also the set-up for each head in our models with partition. As a result, the weights trained from original SIREN and the weights trained from our partitioned-based models have exactly the same keys. Thanks to these settings, in the inference phase we can fine-tune based on our partition methods with the initialized weights trained from the original SIREN.

Results. On top of baseline SIREN, We would demonstrate the effect of our two partition methods both on the inference phase and the training phase. The mean PSNR values for 1 View and 3 View fine-tuning on 300 images are shown in Table 3. We first observe that only fine-tuning based on our partition methods with the initialized weights trained from the original SIREN can lead to better performance than baseline, no matter we use PoG partition or PoS partition. Then we find that the model with PoG partition on both training and inference phase achieves highest PSNR for 3 View fine-tuning, while the model with PoS partition on both training and inference phase achieves highest PSNR for 1 View fine-tuning. Therefore we prove that fine-tuning based on our partition methods with the initialized weight trained from the partition models has the best performance. For detailed performance, we also present a typical example in Figure 6. We can clearly observe that the images obtained from our partition methods contain less noise and more sharper boundaries than the result from the baseline SIREN. More visual results and discussion are attached in Appendix C.

PoS partition as a more flexible choose. As shown in Figure 7, we notice that fine-tuning with PoS method from the initialized weights trained with PoG method would lead to a poor result, while the opposite case still maintain good performance. This phenomenon meets our expectation because each head in PoG method would only learn to fit a regular region and fail to fit an irregular region when fine-tuning with PoS method. On the contrary, the heads in PoS method would learn to fit



(a) Visual results for each step

(b) Ground truth image

(c) Result from SIREN

(d) Result from SIREN ^G_G

(e) Result from SIREN ^S_S

Figure 6: Visual performance of SIREN models with different training and fine-tuning methods. Same abbreviation as in Table 3. We can see the results from our models contain less noise and sharper edges than the baseline SIREN model. (please view this figure in color version.)

(a) Result from SIREN ^G_S

(b) Result from SIREN ^S_G

Figure 7: Performance of fine-tuning with PoS on the PoG trained initialized weight, and its opposite case. The model trained with PoG but fine-tuned with PoS fails, while the model trained with PoS but fine-tuned with PoG achieves good performance. (please view this figure in color version.)

regions with arbitrary shape, including the regular grid. As a result, PoS partition method can be considered as a more flexible partition method than PoG method.

6 Conclusion

In this paper, we investigate the contradiction of learning a continuous function (i.e., a neural network) from a discontinuous signal and demonstrate that the time complexity to force a neural network to fit a discontinuous function is exponentially increasing with the number of high gradients in the input domain, which we call *exponential-increase* hypothesis. We prove that partitioning the input domain into several sub-domains and dedicating smaller neural networks for each sub-domain help to alleviate this contradiction. Based on this observation, we propose two partition methods for learning and learning-to-learn INRs. We also present two partition rules: one is partitioning based on regular grids and the other one is partitioning based on semantic segmentation maps. Our methods significantly speed up the convergence of learning INRs from scratch and also lead to better results for fine-tuning to a new image at fixed step for learning-to-learn INRs.

Finally, future research should consider the effect of *exponential-increase* hypothesis in 3D scene representation, since real 3D scenes in the wild also consist of several discontinuous objects. We hope our findings in the paper can serve as a theoretical support and inspire the follow-on work on learning more powerful implicit neural representations for in-the-wild scenes.

7 REPRODUCIBILITY

We hope everyone can reproduce our work. Therefore we have made the following efforts:

- The detailed settings of all experiments are presented in Section. 5.1 and 5.2.
- The proof for Proposition 1 with full capacity neural networks is presented in Section. 3.1 while the general proof is presented in Appendix.A.
- The formalization of PoS algorithm is presented in Appendix. B.
- More visual results and discussion of partition-based learning-to-learn INRs framework is presented in Appendix.C.
- The LSUN dataset can be downloaded from this open-source link: https://github.com/fyu/lsun.
- Since lots of semantic segmentation maps are included in the paper, we strongly recommend the readers to read this paper in color version.

REFERENCES

Ming-Ming Cheng, Yun Liu, Qibin Hou, Jiawang Bian, Philip Torr, Shi-Min Hu, and Zhuowen Tu. Hfs: Hierarchical feature selection for efficient image segmentation. In *European conference on computer vision*, pp. 867–882. Springer, 2016.

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. *arXiv preprint arXiv:2010.11929*, 2020.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017.

Zekun Hao, Hadar Averbuch-Elor, Noah Snavely, and Serge Belongie. Dualsdf: Semantic shape manipulation using a two-level representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7631–7641, 2020.

Julien NP Martel, David B Lindell, Connor Z Lin, Eric R Chan, Marco Monteiro, and Gordon Wetzstein. Acorn: Adaptive coordinate networks for neural scene representation. *arXiv preprint arXiv:2105.02788*, 2021.

Daniel Meister, Shinji Ogaki, Carsten Benthin, Michael J Doyle, Michael Guthe, and Jiří Bittner. A survey on bounding volume hierarchies for ray tracing. In *Computer Graphics Forum*, volume 40, pp. 683–712. Wiley Online Library, 2021.

Mateusz Michalkiewicz, Jhony K Pontes, Dominic Jack, Mahsa Baktashmotlagh, and Anders Eriksson. Implicit surface representations as layers in neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4743–4752, 2019.

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.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11453–11464, 2021.

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*, pp. 165–174, 2019.

Samuel Pfrommer, Mathew Halm, and Michael Posa. Contactnets: Learning discontinuous contact dynamics with smooth, implicit representations. *arXiv preprint arXiv:2009.11193*, 2020.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

Daniel Rebain, Wei Jiang, Soroosh Yazdani, Ke Li, Kwang Moo Yi, and Andrea Tagliasacchi. Derf: Decomposed radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14153–14161, 2021.

Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14335–14345, 2021.

Peiran Ren, Jiaping Wang, Minmin Gong, Stephen Lin, Xin Tong, and Baining Guo. Global illumination with radiance regression functions. *ACM Trans. Graph.*, 32(4):130–1, 2013.

Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2304–2314, 2019.

Shunsuke Saito, Jinlong Yang, Qianli Ma, and Michael J Black. Scanimate: Weakly supervised learning of skinned clothed avatar networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2886–2897, 2021.

Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *Advances in Neural Information Processing Systems*, 33:20154–20166, 2020.

Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *Advances in Neural Information Processing Systems*, 32, 2019.

Vincent Sitzmann, Eric Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. *Advances in Neural Information Processing Systems*, 33:10136–10147, 2020a.

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, 2020b.

Ivan Skorokhodov, Savva Ignatyev, and Mohamed Elhoseiny. Adversarial generation of continuous images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10753–10764, 2021.

Matthew Tancik, Ben Mildenhall, Terrance Wang, Divi Schmidt, Pratul P Srinivasan, Jonathan T Barron, and Ren Ng. Learned initializations for optimizing coordinate-based neural representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2846–2855, 2021.

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, pp. 641–676. Wiley Online Library, 2022.

Tarun Yenamandra, Ayush Tewari, Florian Bernard, Hans-Peter Seidel, Mohamed Elgharib, Daniel Cremers, and Christian Theobalt. i3dmm: Deep implicit 3d morphable model of human heads. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12803–12813, 2021.

Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

A COMPLEXITY OF DEDICATING SMALLER NEURAL NETWORKS

In this part, we show that with larger k, proposition 1 can be generalized to the case of fitting each sub-domain with smaller neural networks, whose complexity of fitting one boundary is larger than p.

Proof: Since we dedicate smaller neural networks for each sub-domain, we can assume the complexity of fitting one boundary for each smaller neural network is $\{p_1, p_2, ..., p_k\}$, where $p_i > p$. Then the total complexity of parallelly fitting all sub-domain with separate neural networks is

$$p_1^{n_1} + p_2^{n_2} + ... + p_k^{n_k}. \tag{10}$$

Defining $\hat{n} = \max(n_1, n_2, ..., n_k)$ and $\hat{p} = \max(p_1, p_2, ..., p_k)$, we hold inequations 11:

$$\frac{p_1^{n_1} + p_1^{n_2} + ... + p_k^{n_k}}{\hat{p}^{\hat{n}}} = \frac{\hat{p}^{n_1}}{\hat{p}^{\hat{n}}} + \frac{\hat{p}^{n_2}}{\hat{p}^{\hat{n}}} + ... + \frac{\hat{p}^{n_k}}{\hat{p}^{\hat{n}}} \leq 1 + 1 + ... + 1 = k. \tag{11}$$

Empirically, we should optimize each neural network for several times, so we have $p^{n_i} \geq 2$, then the following inequation hold:

$$\frac{p^n}{\hat{p}^{\hat{n}}} = \frac{p^{(n_1+n_2+...+n_k)}}{\hat{p}^{\hat{n}}} = \frac{p}{\hat{p}} \prod_{n_i \neq \hat{n}} p^{n_i} \geq \frac{p}{\hat{p}} \cdot 2^{k-1}. \tag{12}$$

As long as k is large enough so that $\frac{p}{\hat{p}} \cdot 2^{k-1} > k$, proposition 1 is proved for smaller neural networks.

B FORMALIZATION FOR POS ALGORITHM

The detailed implementation of PoS algorithm is presented as below:

Algorithm 1 Partition Based on Semantic Segmentation Maps

Input: An image that needed to be partitioned, I; the number of required sub-domains k
Output: The semantic segmentation map, M;

1: Delivering Hierarchical Feature Selection algorithm on I and obtaining initialized segmentation map M_i;
2: Delivering connected components algorithm to re-label those unconnected parts on M_i;
3: **while** Number of sub-domain(M_i) $>k$ **do**
4: Finding the sub-domain D_s in M_i which contains smallest area of region;
5: Finding the sub-domain D_r who is the neighbor of D_s and contains smallest area of region;
6: Changing the label of D_s to the label of D_r
7: **end while**
8: **return** M_i;

C MORE RESULTS AND DISCUSSION FOR LEARNING-TO-LEARN INRS

We explore which model achieves highest 3 Views PSNR value on all 300 images and present the results in Table 4. We find that the $\text{SIREN}_{\text{G}}^{\text{G}}$ model achieves best performance in most cases, while $\text{SIREN}_{\text{S}}^{\text{S}}$ achieves best performance in the rest cases. We think this is because training a good initialized weight that is suitable for arbitrary shape of sub-domain (PoS) is much harder than training a good initialized weight that is suitable for regular grid (PoG).

Table 4: Times of best 3 Views performance on 300 images for all models.

Architectures	$\text{SIREN}_{\text{G}}^{\text{G}}$	$\text{SIREN}_{\text{S}}^{\text{S}}$	$\text{SIREN}_{\text{G}}^{\text{S}}$	SIREN_{G}	SIREN_{S}	SIREN	$\text{SIREN}_{\text{S}}^{\text{G}}$
# highest PSNR	273	17	0	0	0	0	0

We also present more examples that our partition-based models defeat the baseline in Figure 8 and 9. We can clearly see that our partition-based models generate less noise than baseline. And the partition-based models also generate sharper discontinuous boundaries between two separate objects in the scene, which further proves our Proposition 1.

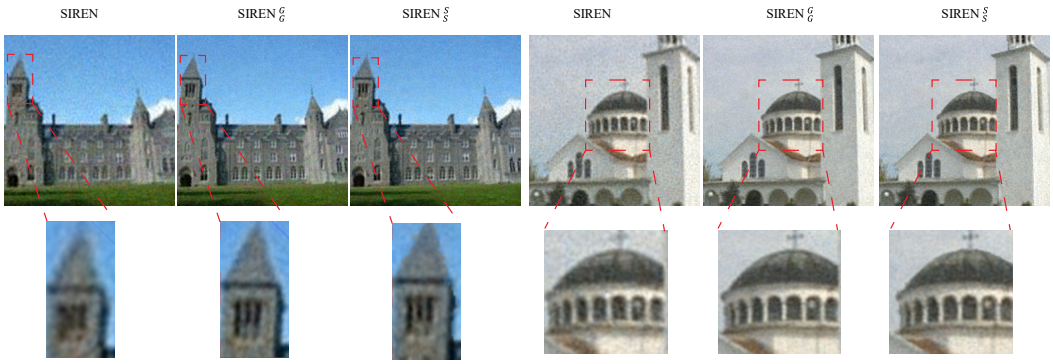


Figure 8: Performance comparison of 3 Views results of baseline model and our partition-based models. Our partition-based model would generate less noise as well as sharper boundaries.

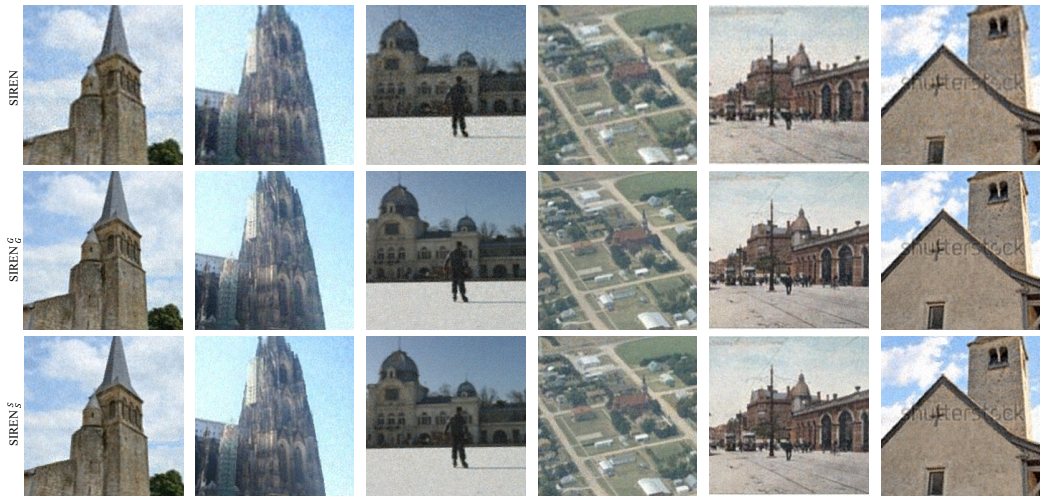


Figure 9: Performance comparison of 3 Views results of baseline model and our partition-based models. Our partition-based model would generate less noise as well as sharper boundaries.