VERSATILE NEURAL PROCESSES FOR LEARNING IMPLICIT NEURAL REPRESENTATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Representing a signal as a continuous function parameterized by neural network (a.k.a. Implicit Neural Representations, INRs) has attracted increasing attention in recent years. Neural Processes (NPs), which model the distributions over functions conditioned on partial observations (context set), provide a practical solution for fast inference of continuous functions. However, existing NP architectures suffer from inferior modeling capability for complex signals. In this paper, we propose an efficient NP framework dubbed Versatile Neural Processes (VNP), which largely increases the capability of approximating functions. Specifically, we introduce a bottleneck encoder that produces fewer and informative context tokens, relieving the high computational cost while providing high modeling capability. At the decoder side, we hierarchically learn multiple global latent variables that jointly model the global structure and the uncertainty of a function, enabling our model to capture the distribution of complex signals. We demonstrate the effectiveness of the proposed VNP on a variety of tasks involving 1D, 2D and 3D signals. Particularly, our method shows promise in learning accurate INRs w.r.t. a 3D scene without further finetuning.

1 INTRODUCTION

A recent line of research on learning representations is to model a signal (*e.g.*, image, 3D scene) as a continuous function that map the input coordinates into the corresponding signal values. By parameterizing a continuous function with neural networks, such implicitly defined representations, *i.e.*, implicit neural representations (INRs), offer many benefits over conventional discrete (*e.g.*, grid-based) representations, such as the compactness and memory-efficiency (Sitzmann et al., 2020b; Tancik et al., 2020; Mildenhall et al., 2020; Chen et al., 2021a). Characterizing/parameterizing a signal by a corresponding set of network parameters generally requires re-training the neural network, which is computationally costly. In practice, at test time, it is desired to have models that support fast adaptation to partial observations of a new signal without finetuning.

In fact, the Neural Processes (NPs) family (Jha et al., 2022) supports such merit. It meta-learns the implicit neural representations of a probabilistic function conditioned on partial signal observations. During test-time inference, it enables the prediction of the function values at target points within a single forward pass. Naturally, given partial observations of a signal, there exists uncertainty inside its continuous function since there are many possible ways to interpret these observations (*i.e.*, context set). The NP methods (Garnelo et al., 2018a;b) learn to map a context set of observed input-output pairs to a conditional distribution over functions (with uncertainty modeling). However, it has been observed that NPs are prone to underfit the data distribution. Following the spirits of variational auto-encoders (Kingma & Welling, 2014), the work of (Garnelo et al., 2018b) introduces a global latent variable to better capture the uncertainty in the overall structure of the function, which still suffers from the inferior capability for modeling complex signals. Attentive Neural Processes (ANP) (Kim et al., 2019) can further alleviate this issue, which leverages the permutation-invariant attention mechanism (Vaswani et al., 2017) to reweight the context points and the target predictions. However, taking each context point as a token, ANP has troubles in processing complex signals that requires abundant context points as condition (*e.g.*, image with high resolution), where the computational cost is very expensive. Moreover, for complex signals, modeling the global structure and uncertainty of the function with a single latent Gaussian variable may be suboptimal. It is worthwhile to explore an efficient framework to excavate the potential of NPs in modeling complex signals.

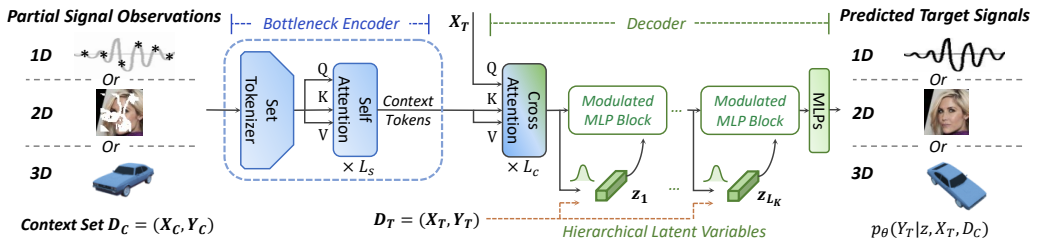


Figure 1: The architecture of the proposed Versatile Neural Processes. It consists of a bottleneck encoder and a hierarchical latent modulated decoder. At the bottleneck encoder, we encode the input context set into fewer and informative context tokens by a set tokenizer followed by self-attention blocks, which enable tolerable computational complexity and high network capability. The decoder consists of cross-attention modules, and our proposed multiple modulated MLP modules, which enhance the model expressiveness to complex signals.

In this paper, we propose Versatile Neural Processes (VNP), an efficient and flexible framework for meta-learning of implicit neural representations. Figure 1 shows the framework of VNP. Specifically, VNP consists of a bottleneck encoder and a hierarchical latent modulated decoder. The bottleneck encoder powered by the set tokenizer and self-attention blocks encodes the set of context points into fewer and informative context tokens, refraining from high computational cost especially on complex signals while attaining higher modeling capability. At the decoder, we hierarchically learn multiple latent Gaussian variables for jointly modeling the global structure and uncertainty of the function distributions. Particularly, we sample from the latent variables and use them to modulate the parameters of the MLP modules. Our VNP has high expressiveness to complex signals (*e.g.*, 2D images and 3D scenes) and significantly outperforms existing NPs approaches on 1D synthetic data.

We summarize our main contributions as below:

- We propose Versatile Neural Processes (VNP) that is capable of learning accurate INRs for approximating the function of a complex signal.
- We introduce a bottleneck encoder to produce compact yet representative context tokens, facilitating the processing of complex signals with tolerable computational complexity.
- We design a hierarchical latent modulated decoder that can better capture and describe the structure and uncertainty of function through the joint modulation from the multiple global latent variables.
- We implement the VNP framework on 1D, 2D, and 3D signals respectively, demonstrating the state-of-the-art performance on a variety of tasks. Particularly, our method shows promise in learning accurate INRs of 3D scenes without further finetuning.

2 Related Work

Implicit Neural Representations (INRs). INRs aim at parameterizing a signal by a differentiable neural network, *i.e.*, learning a *continuous* mapping function *w.r.t.* the signal (Stanley, 2007; Sitzmann et al., 2020b; Tancik et al., 2020). In the seminal work CPPN (Stanley, 2007), a neural network is trained to learn the implicit function that fits a signal, *e.g.*, an image. Given any spatial position identified by a 2D coordinate, the model that acts as a function, outputs the color value of this position. Such continuous representations, as a powerful paradigm, have a wide range of applications such as 2D image super-resolution (Chen et al., 2021b), modeling shapes (Chen & Zhang, 2019; Park et al., 2019) and textures (Oechsle et al., 2019), and 3D scene reconstruction (Mildenhall et al., 2020; Martin-Brualla et al., 2021; Niemeyer & Geiger, 2021). Most of these methods require re-training the neural network to model/overfit a new signal, which is computationally costly (Sitzmann et al., 2020b; Tancik et al., 2020; Mildenhall et al., 2020; Chen et al., 2021a).

In practice, it is desired to have models that support fast adaptation to a new signal, *i.e.*, approaching the continuous function of this signal without abundant steps of optimization in inference. Some

works (Chen et al., 2021b; Lee et al., 2021; Sitzmann et al., 2020a) adopt standard gradient-based meta-learning algorithms to learn the initial weight parameters of the network (Finn et al., 2017). However, they still require a few gradient decent steps to fit for the new signals.

Neural Processes (NPs). Neural Processes actually can learn the continuous function conditioned on partial observations of a signal, enabling fast adaptation to a new signal (not requiring finetuning in inference). The series of NPs methods (Garnelo et al., 2018a;b) provide probabilistic solutions in predicting continuous functions from partial observations. NPs approximate the distributions in the function space, which formulates Stochastic Processes (Ross et al., 1996), by introducing stochasticity in function realization. A line of researches on neural processes has been introduced, targeting at improving the prediction accuracy (Kim et al., 2019; Lee et al., 2020; Wang & Van Hoof, 2020; Volpp et al., 2021), preserving the stationarity of stochastic processes (Gordon et al.; Foong et al., 2020), and generalizing to observation noise (Kim et al., 2022). The work of Neural Processes (Garnelo et al., 2018b) learns a latent variable distribution to capture the global uncertainty in the overall structure of the function, which is optimized with variational inferece (Kingma & Welling, 2014). Attentive Neural Processes (ANP) leverages the attention mechanism to enhance the representation of each context point and alleviate the underfitting problem (Kim et al., 2019). Transformer Neural Processes (TNP) (Nguyen & Grover, 2022) similarly takes each context point as a token and leverages transformer architecture to approximate the function. However, for complex signals that requiring abundant context points as condition (*e.g.*, image with high resolution), the computational complexities of ANP and TNP are very high which are quadratic with respect to the number of context points. It is desired to have a framework that can effectively approximate the functions of complex signals. In this work, we introduce a strong NP framework, Versatile Neural Processes (VNP), which leverages informative context tokens and explores the hierarchical global latent variables for modulation, leading to superior approximation of function distributions.

3 Revisiting the Problem Formulation of NPs

Neural Processes (NPs) (Garnelo et al., 2018b;a) are a class of methods that approximate the probabilistic distribution of the continuous function conditioned on partial observations. Suppose we have a labeled context set $D_C = (X_C, Y_C) := (\mathbf{x}_i, \mathbf{y}_i)_{i \in C}$ sampled from a continuous function with inputs $\mathbf{x}_i$ and outputs $\mathbf{y}_i$, NPs target at predicting an arbitrary and finite number of target points $D_T = (X_T, Y_T) := (\mathbf{x}_i, \mathbf{y}_i)_{i \in T}$ by learning the input-output mapping function f. Given some observations (a set of context points) of a signal, many possible functions may match well to these observations and thus naturally there exists uncertainty on the functions. The conditional distributions of targets points can be modeled as:

$$p_\phi(Y_T | X_T, D_c) = \prod\nolimits_{(\mathbf{x},\mathbf{y}) \in D_T} \mathcal{N}(\mathbf{y}; \mu_\mathbf{y}(\mathbf{x}, D_c), \sigma^2_\mathbf{y}(\mathbf{x}, D_c)). \tag{1}$$

To generate coherent function predictions and better model the function distributions, Garnelo *et al.* (Garnelo et al., 2018b) introduce the (Latent) Neural Process by encoding the global structure and uncertainty of the function into a latent Gaussian variable via Bayesian inference:

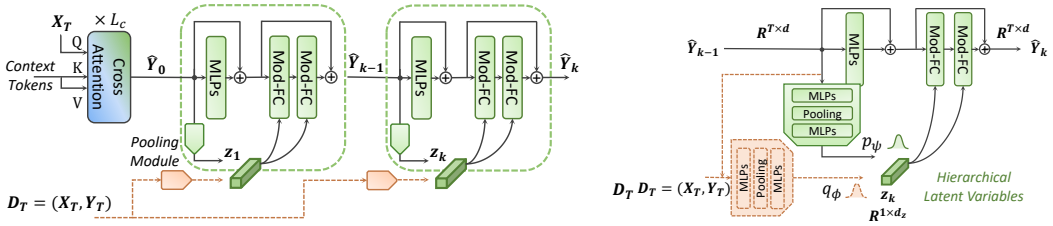
$$\mathbf{z} \sim p_\theta(\mathbf{z}|X_T, D_C); \quad p_{\phi,\theta}(Y_T|X_T, \mathbf{z}) = \prod\nolimits_{(\mathbf{x},\mathbf{y}) \in D_T} \mathcal{N}(\mathbf{y}; \mu_\mathbf{y}(\mathbf{z}, X_T, D_C), \sigma^2_\mathbf{y}(\mathbf{z}, X_T, D_C)). \tag{2}$$

Due to the intractable log-likelihood, some previous works adopt amortized variational inference (Kingma & Welling, 2014), which is also used to optimize our proposed framework. We can derive the evidence lower bound (ELBO) on $\log p_\theta(Y_T|X_T, \mathbf{z})$, where the ELBO can be viewed as a combination of the reconstruction term (the first term) and the KL term (the second term):

$$\mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z}|D_T)}[\log p_\theta(Y_T|\mathbf{z}, X_T, D_C) - D_{KL}[q_\phi(\mathbf{z}|D_T)||p_\psi(\mathbf{z}|X_T, D_C)]]. \tag{3}$$

Here, ψ and θ refer to the parameters of conditional prior encoder and the decoder, similar to conditional VAEs (Sohn et al., 2015; Ivanov et al., 2019). $q_\phi(\mathbf{z}|D_T)$ denotes the posterior distribution of the latent variable given the ground truth target points, which is only used in training and not accessed in inference.

So far, we have revisited the problem formulation of NPs. Since NPs approximate the distribution over functions with some function observations, they can also be regarded as *probabilistic, continuous, conditional* generative models, where the generation result is a continuous function.



(a) Pipeline of the decoder. (b) Modulated MLP block.

Figure 2: Diagram of the decoder with the hierarchical latent modulated MLPs. .

4 Versatile Neural Processes

We propose an efficient NP framework, Versatile Neural Processes (VNP), which provides much improved capability in approximating the function of a various signal. Figure 1 provides an overview of our proposed VNP. VNP is generic that can be used to generate the functions for 1D, 2D or 3D signals. For test-time inference, the inputs are a set of context points $D_C = (X_C, Y_C)$ and the coordinates of target points X_T. The outputs are the predicted values of target points Y_T. The continuous function is parameterized by the network weights. The proposed VNP consists of a bottleneck encoder that efficiently encodes the context points into fewer informative context tokens, refraining from high computational cost especially on complex signals. At the decoder side, cross-attention layers are utilized to exploit the contexts relevant to the given target followed by a stack of modulated MLP blocks. Particularly, we introduce multiple global latent variables to jointly modulate the MLP parameters, which facilitates the modeling of complex signal.

4.1 Bottleneck Encoder in VNP

Some NP methods encode the context points to produce local feature representations at target coordinates. The early work (Garnelo et al., 2018b) uses simple MLP layers to learn features of the context set and suffers from underfitting problem. Attentive NP (Kim et al., 2019) and Transformer NP (Nguyen & Grover, 2022) enhances the feature representation by introducing point-wise self-attention with each context point taken as a token. However, when the number of context points is large (*e.g.*in order to model an image signal with many details), the computational burden is heavy and such a framework is impractical.

We address this issue by simply introducing a set tokenization module (set tokenizer) followed by self-attention layers. The set tokenizer is instantiated by a set convolution layer (Zaheer et al., 2017) which transforms the neighboring context points to a token. Taking a 2D image as an example, with the kernel size of $k \times k$ and stride of k, the set tokenizer can reduce the number of sample points by k^2 times (assume the image resolution is an integer multiple of k). By using the set tokenzier, the complexity of attention layers is reduced from the first term to the second term as below:

$$\mathcal{O}(L_s N_C^2 + L_c N_C N_T) \rightarrow \mathcal{O}(L_s \frac{N_C^2}{k^4} + L_c \frac{N_C}{k^2} N_T), \tag{4}$$

where L_s and L_c are the number of the self-attention layers and the cross-attention layers, N_C and N_T are the number of context points and target points. The difference between the set convolution and the conventional convolution is that the former can handle missing points (*e.g.*, in the application of inpainting) and the data that live "off the grid" (*e.g.* time series data that observed irregularly at any time). The combination of set tokenizer and attention would empower our framework the flexibility in processing 3D signals, which is infeasible for ConvNP (Gordon et al.; Foong et al., 2020) because ConvNP requires to preserve all the 3D grids, which is very inefficient in sparse 3D space.

4.2 Hierarchical Latent Modulated MLPs

In the previous NP methods (Garnelo et al., 2018b; Kim et al., 2019), they learn a *single* global latent Gaussian variable from the observed context set to model the distributions of a function. However, it potentially limits the expressiveness of the model. Intuitively, increasing the dimension of the latent variables may increase such flexibility, but in practice, this is not sufficient (Lee et al., 2020).

We design a hierarchical latent modulated decoder in order to better model the complex distribution of a function. We sequentially learn multiple global latent variables to modulate the parameters of MLP blocks. Figure 2 shows the details of the designed hierarchical structure.

The decoder consists of L_c cross-attention blocks and L_K modulated MLP blocks. With the target location X_T as query and the context tokens from encoder as the keys and values, the cross-attention blocks output the target location features $\hat{Y}_0 \in \mathbb{R}^{T \times d}$, where T is the number of target coordinates and d is the feature dimension. The sequential modulated MLP blocks enable the exploitation of hierarchical global latent variables for the approximation of a complex signal. The output of the final (L_K^{th}) modulated MLP block goes through two MLP layers to estimate the probability of Y_T.

Figure 2b illustrates the detailed network structure of a modulated MLP block. We build a Modulated MLP block by stacking two modulated MLP layers and two unmodulated MLP layers. This block refines $\hat{Y}_{k-1}$ to output features $\hat{Y}_k$, which is the input of the next Modulated MLP block. At the heart of each block is a latent variable $\mathbf{z} \in \mathbb{R}^{1 \times d_{\mathbf{z}}}$ modeled by a Gaussian distribution. The generation process of $\mathbf{z}_k$ (of the kth block) can be formulated as follows:

$$p_\psi(\mathbf{z}_k|\hat{Y}_{<k}, D_C, X_T) = \mathcal{N}(\mu_{\mathbf{z_k}}, \sigma^{\mathbf{2}}_{\mathbf{z_k}}) \leftarrow \text{MLPs}\left(\text{AvgPool}\left(\text{MLPs}(\hat{Y}_{k-1})\right)\right), \tag{5}$$

where MLPs refer to two MLP layers and an intermediate ReLU activation layer. The prediction results Y_{k-1} from the previous block (*i.e.*, the $(k-1)^{th}$ block) are used for calculating the conditional prior distribution of $\mathbf{z}_k$, *i.e.*, $p_\psi(\mathbf{z}_k|\hat{Y}_{<k}, D_C, X_T)$. During training, the conditional posterior distribution $q_\phi(\mathbf{z}_k|\hat{Y}_{<k}, D_C, D_T)$ can be calculated as well, by incorporating the ground truth target signal values $\mathrm{Y_T}$:

$$q_\phi(\mathbf{z}_k|\hat{Y}_{<k}, D_C, D_T) = \mathcal{N}(\mu_{\mathbf{z_k}}, \sigma^{\mathbf{2}}_{\mathbf{z_k}}) \leftarrow \text{MLPs}\left(\text{AvgPool}\left(\text{MLPs}([\hat{Y}_{k-1}, D_T])\right)\right). \tag{6}$$

As marked with dashed lines in our figures, Y_T only participates in training and cannot be accessed during inference.

After sampling the low-dimensional latent variable $\mathbf{z}_k$, we use the modulated fully-connected (ModFC) layer (Karras et al., 2020; 2021) to adjust the parameters of modulated MLP layers, taking a sampled realization of $\mathbf{z_k}$ as the style vector input. Unlike previous NP methods that concatenate the latent variable to every target coordinate, modulating the MLP parameters from the low-dimensional latent variables is a more flexible way to represent continuous functions (Sitzmann et al., 2020b). Please see Appendix A for more details for the mechanism of the modulated MLP.

We model the function representations by multiple latent variables $(\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_{L_K})$. The KL term in Eq. 3, which measures the mismatch between the approximated distribution and ground truth distribution, takes the hierarchical format as

$$D_{KL} = \sum_{k=1}^{L_K} D_{KL}[q_\phi(\mathbf{z}_k|\hat{Y}_{<k}, D_C, D_T)||p_\psi(\mathbf{z}_k|\hat{Y}_{<k}, D_C, X_T)]. \tag{7}$$

This decomposed KL term together with the original reconstruction term formulates our final optimization objective as

$$\mathcal{L} = \mathbb{E}_{\mathbf{z}_{1:L_K} \sim q_\phi(\mathbf{z}_{1:L_K}|D_T)}[-\log p_\theta(Y_T|\mathbf{z}_{1:L_K}, X_T, D_C) + \beta \cdot D_{KL}], \tag{8}$$

where β denotes a weight to balance the importance between the two terms. In our experiments on 2D and 3D signals, we multiply the KL term with a small weight β to better capture the uncertainty of function distributions (Higgins et al.). In addition, we emphasize that although there are some prior works designing hierarchical architecture for VAEs (Sønderby et al., 2016; Vahdat & Kautz, 2020; Child, 2021) or double latent variable models for NP (Wang & Van Hoof, 2020), our proposed hierarchical architecture is different in principle. Here, every latent variable is a low-dimensional vector obtained after average pooling. Therefore, the designed hierarchical architecture can deal with arbitrary target coordinates and thereby can be used for boosting the performance of approximating the global structure of continuous functions.

5 Experiments

The proposed Versatile Neural Process (VNP), as an efficient meta-learner of implicit neural representations, can be implemented into a variety of tasks. We evaluate the effectiveness of VNP on 1D function regression (subsection 5.1), 2D image completion and superresolution (subsection 5.2), and view synthesis for 3D scenes (subsection 5.3), respectively.

	RBF kernel GP		Matern kernel GP		Parameters
	context	target	context	target	
CNP	1.023±0.033	0.019±0.015	0.935±0.036	-0.124±0.010	0.99 M
BANP	1.380±0.000	0.267±0.001	1.380±0.002	0.072±0.002	1.58 M
ConvNP	1.382±0.001	0.275±0.001	1.383±0.001	0.081±0.008	1.97 M
Stacked ANP	1.381±0.001	0.400±0.004	1.381±0.001	0.183±0.012	1.52 M
Stacked ANP +	1.381±0.001	0.406±0.006	1.381±0.001	0.188±0.014	2.31 M
HNP	1.374±0.002	0.561±0.003	1.377±0.001	0.336±0.008	1.66 M
HNP-Mod	1.379±0.001	0.627±0.020	1.371±0.004	0.370±0.023	1.96 M
VNP	1.377±0.004	**0.651±0.001**	1.376±0.004	**0.439±0.007**	2.29 M

Table 1: The test log likelihood (larger is better) on the synthetic 1D regression experiment. The proposed VNP outperforms previous methods by a large margin. Both the pre-processing transformer and the hierarchical structure improve the expressiveness of function representations. Stacked ANP + means the enhanced version of Stacked ANP with more channels.

5.1 1D Signal Regression

We implement the proposed VNP to learn the implicit neural representations for 1D signal regression. This classical 1D regression aims at predicting the function values at given target locations, conditioned on several observations of the samples from the function. Following (Kim et al., 2019), we measure the performance by considering the context set likelihood and target set likelihood, which reflects the context reconstruction error and target prediction error, respectively.

Settings. We train the models on synthetic functions drawn from prior function distributions synthesized with different kernels (RBF, Matern) by following (Gordon et al.; Kim et al., 2022). For the evaluated methods, we employ importance weighted sampling (Burda et al., 2016) to evaluate the log likelihood, where the last four methods in Table 1 are measured by sampling latent variables from the posterior distribution and calculating the tighter ELBO with importance weighted sampling. For fair comparison, we manage to keep the network size comparable with that of other methods, which is completed by adjusting channel number. Please refer to Appendix B for detailed settings.

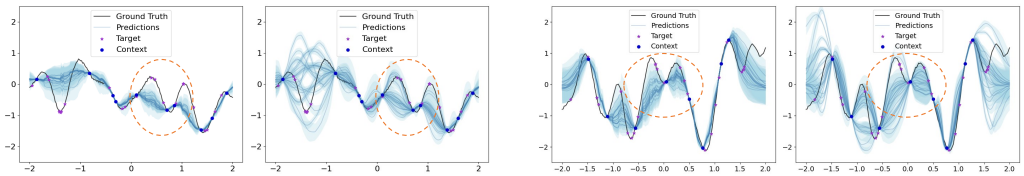
Comparison with the State-of-the-Arts. We compare our method with conditional neural process (CNP) (Garnelo et al., 2018a), (stacked) attentive neural process (ANP) (Kim et al., 2019) (with stacked self-attention layers), bootstrapping attentive neural process (BANP) (Lee et al., 2020), and convolutional neural process (ConvNP) (Foong et al., 2020). Table 1 shows the results in terms of log likelihood. Most of the models can provide satisfactory reconstruction results for context points, except CNP which suffers from underfitting problem (Garnelo et al., 2018b). On the context points, ours also achieves comparable performance. However, the prediction results of other methods on the unseen target points are much worse than that on the context sets. In contrast, our final model, VNP, outperforms previous approaches by a large margin at the target points on both of the function distributions. Note that although BANP attempts to increase the expressiveness of function representations by using latent variable bootstrapping from the perspective data resampling.

Ablation Study. We conduct a group of ablation study on this 1D regression task to investigate the effectiveness of different components. Based on ANP, we first build a Hierarchical Neural Process (HNP) with hierarchical latent variables. Note that similar to ANP, the implemented HNP still concatenates every target coordinate with the global latent variables. We observe significant improvements in Table 1 by comparing HNP with ANP, demonstrating that the hierarchical global latent variables boost the performance of function approximation. Secondly, the modulated MLP layer for implicit function parameterization can be further equipped in HNP, referred to as HNP-Mod. Compared with the concatenation of latent variable, using low-dimensional latent variables to modulate the network parameters enables more flexible function approximating. Based on HNP-Mod, we finally add the bottleneck encoder to learn conditional network inputs that are adaptive to target locations, building our final model VNP. In Table 1, we can find this powerful pre-processing encoder can further improve the performance, because the set tokenizer (set convolution here) can preserve locality to learn more appropriate features.

In Table 2, we further ablate on the detailed hierarchical structures in our model. As we can observe, when we increase the number of Modulated MLP blocks, the prediction performance (at target points)

	number of Modulated MLP blocks					
	0	2	4	6	8	6 / single $\mathbf{z}$
context	1.375±0.001	1.376±0.001	1.376±0.001	1.376±0.001	1.376±0.001	1.376±0.001
target	0.076±0.001	0.336±0.028	0.371±0.001	**0.439±0.007**	0.435±0.028	0.245±0.003

Table 2: Ablation study about the detailed hierarchical structure on Matern kernel. Here we also provide the results of test log-likelihood (larger is better).



(a) Matern kernel. ANP vs. our VNP.

(b) RBF kernel. ANP vs. our VNP.

Figure 3: Visualizations of the 1D regression results. VNP delivers diverse function prediction results, while ANP (Kim et al., 2019) tends to underestimate the function variances at some target locations.

is improved until saturated with 6 blocks (*i.e.*, $L_K = 6$). In addition, the column with "6 / single $\mathbf{z}$" means we use exactly the same network structures as $L_K = 6$, but instead only impose the KL constraint to the final latent variable $\mathbf{z}_6$. It is found that the performance will also drop dramatically, which verifies that the gains come from the hierarchical design instead of the increased network capacity. This ablation study guides us to set the number of Modulated MLP blocks as 6 to compare with other methods, which can keep a balance between the performance and the complexity.

Visualizations. We visualize the obtained function distributions from stacked ANP (Kim et al., 2019) and our VNP, by sampling the latent variables 20 times. Given partial observations of a signal, there exists uncertainty on the continuous function since there are many possible ways to interpret these observations (i.e., context set). An excellent Neural Process model should be able to model such uncertainty through the fitted functions. In other words, the generated functions conditioned on the context set should approximate the data distribution and cover the target set points. As shown in Figure 3, we can see that the generated functions from the stacked ANP cannot predict the target points accurately, *e.g.*, those regions marked by orange. In contrast, our method provide good approximations for the continuous functions, where the distribution covers the groundtruth function.

5.2 2D Images Completion and Superresolution

A 2D image can be modeled by a continuous function that maps the 2D pixel coordinates to the color values. We implement our VNP framework for learning the continuous function of 2D images, which supports tasks such as image completion and super-resolution to arbitrary size.

Settings. We conduct experiments on CelebA dataset (Liu et al., 2015), mainly with the resized resolution of 64×64. Due to the limited representation capability or the high requirement on computation resources, most previous NP methods (Kim et al., 2019; Lee et al., 2020) are in general trained and evaluated on relatively low resolution such as 32×32. Our framework enables the processing of complex signal with higher resolution. We train a single model that supports image completion and super-resolution tasks. During training, we control the context ratio as 0.03, which means 3% pixels are taken as context set. The target ratio is set as 0.15 and the target set has partial intersection with the context set. More detailed experimental settings can be found in Appendix B.

Visualizations. We compare our VNP with stacked ANP (Kim et al., 2019) tested with three different context ratios respectively as shown in Figure 4. Our VNP generates diverse and realistic image completion results with fine details. In comparison, the results of Stacked ANP are blurred.

In Figure 5, we show that the proposed VNP achieves satisfactory results for image completion and superresolution on CelebA64 dataset. As a unified framework, with a single model, it can support image completion, superresolution, and the concurrent of completion and superresolution.

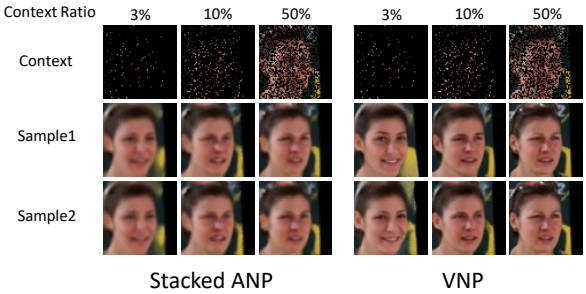


Figure 4: Qualitative comparisons with Stacked ANP (Kim et al., 2019). Our VNP presents diverse and realistic image completion results.

CelebA64	NLL (lower is better) context ratio = 0.03
Stacked ANP	2.988
VNP, $ks = 1$, $L_K = 1$	2.994
VNP, $ks = 1$, $L_K = 6$	2.953
VNP, $ks = 2$, $L_K = 6$	2.952
VNP, $ks = 4$, $L_K = 6$	2.964

Table 3: Quantitative results measured by Eq.8 on the test set. Lower is better. ks is the kernel size in set tokenizer. L_K is the number of Modulated MLP blocks.

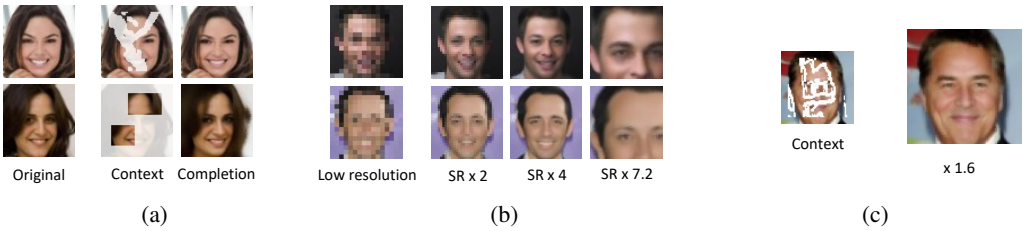


Figure 5: Visualization of VNP results on CelebA64 (Liu et al., 2015). (a) Image Completion. (b) Superresolution to arbitrary size. (c) Handling image completion and superresolution simultaneously.

GFLOPs	CelebA64			CelebA178		
	Context Ratio			Context Ratio		
	0.05	0.25	1.0	0.05	0.25	1.0
Stacked ANP	17.1	39.8	198.5	190.0	867.2	12150
Our VNP	27.6	27.6	27.6	204.5	204.5	204.5

Table 4: Comparing our VNP with stacked ANP (Kim et al., 2019) about the complexity, measure by FLOPs. The bottleneck encoder plays an important role in reducing the computational cost. The statistic in gray is estimated since we have an upper bound limitation of GPU memory.

Influence of Patch/Kernel Size. We conduct an ablation study to investigate the influence of the patch/kernel size in our set tokenizer. We set the stride the same as the kernel size. The results are shown in Table 3. It is observed that using larger kernel size does not degrade the performance obviously. In addition, using hierarchical structure brings gains.

Complexity Comparisons. Thanks to our bottleneck encoder design, the proposed VNP is a practical framework for complex signal modeling, where the number of context points as condition is usually large. For comparison, we calculate the GFLOPs of stacked ANP and our VNP when target ratio is 0.25 on CelebA64 and CelebA178 respectively. Here, the kernel size of set tokenizer is 4 on CelebA64 and 10 on CelebA178. The computational complexity comparisons in terms of GFLOPs are shown in Table 4. Since we use set tokenizer to reduce the number of tokens and process the context set in image grids, the GFLOPs of VNP would be smaller than that of Stacked ANP, especially when the context ratio is high.

5.3 View Synthesis on 3D Scene

Implicit neural representations excel at representing 3D signals. However, since 3D signals are usually much more complex, most previous works (Mildenhall et al., 2020; Martin-Brualla et al., 2021) require expensive optimization to fit the neural networks to represent the scenes. We focus on the task of view synthesis in this section to evaluate our proposed VNP.

(a) Cars. (b) Chairs.

Figure 6: Novel view synthesis on ShapeNet (Chang et al., 2015) Objects. Our VNP presents more realistic prediction results than the blurry results of NeRF-VAE (Kosiorek et al., 2021).

		PSNR (dB)		
		Cars	Lamps	Chairs
Deterministic	Learned Init (Chen et al., 2021b)	22.80	22.35	18.85
	Trans-INR (Chen & Wang, 2022)	23.78	22.76	**19.66**
Probabilistic	NeRF-VAE (Kosiorek et al., 2021)	21.79	21.58	17.15
	Our VNP	**24.52**	**24.07**	19.32

Table 5: The quantitative results of one-shot novel view synthesis. We compare the proposed VNP with both deterministic and probabilistic methods. All of these methods are able to produce INRs representing the 3D scenes within few steps.

Settings. We follow the spirits of NeRF (Mildenhall et al., 2020) that fits a network to map the world coordinate into the corresponding RGB values and volume density. We use bottleneck encoder (tokenization from the image patches) to calculate the adaptive input of the MLPs, queried by target world coordinate. Then the hierarchically learned global latent variable will modulate the parameters of MLPs to predict the RGB values and volume density. Note that there is an extra volume rendering process inside each decoding block before the pooling module, because it requires transferring the world coordinate to the image coordinate to compute the latent distribution of $\mathbf{z}_k$. We conduct experiments on ShapeNet (Chang et al., 2015) objects, including three sub-dataset: cars, lamps, and chairs. More details on the network structures and hyper parameters can be found in Appendix B.

Comparisons. Our VNP model can generate probabilistic implicit neural representations for previous unseen scene in a single forward pass. In this 3D task, one prior work NeRF-VAE (Kosiorek et al., 2021) can also *generate* the scene from the randomly sampled latent variable. We make quantitative and qualitative comparisons with NeRF-VAE. In addition, we compare with another two works, including (Chen et al., 2021b) and (Chen & Wang, 2022) that learn implicit neural representations for previous unseen scene using a few steps, although they are not in the family of probabilistic models and thus cannot model distributions of function from priors. As shown in Table 5, our method achieves the best performance in terms of Peak Signal-to-Noise Ratio (PSNR) on Cars and Lamps and comparable performance with Trans-INR (Chen & Wang, 2022) on Chairs. The qualitative comparisons in Figure 6a show that our method VNP produces much better prediction of the novel view than NeRF-VAE. Note that during the optimization, VAE estimates the distribution over the dataset given a sample. In contrast, NPs estimate the conditional distribution over the labels of the target points given the set of labeled context points, which is more adaptive to the signal.

6 Conclusion and Discussion

The Neural Processes family provides an efficient way for learning implicit neural representations by approximating the function distribution when only partial observation of a signal is given. We propose an efficient NP framework, Versatile Neural Processes (VNP), that largely increases the capability of approximating functions. Our bottleneck encoder and hierarchical latent modulated decoder enables strong modeling capability to the complex signals. Through comprehensive experiments, we show the effectiveness of the proposed VNP on 1D, 2D and 3D signals. Our work demonstrates the potential of neural process as a promising solution for efficient learning of INRs in complex 3D scenes.

REFERENCES

Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. In *4th International Conference on Learning Representations, ICLR 2016*, 2016.

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser Nam Lim, and Abhinav Shrivastava. Nerv: Neural representations for videos. volume 34, 2021a.

Yinbo Chen and Xiaolong Wang. Transformers as meta-learners for implicit neural representations. *arXiv preprint arXiv:2208.02801*, 2022.

Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8628–8638, 2021b.

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*, pp. 5939–5948, 2019.

Rewon Child. Very deep vaes generalize autoregressive models and can outperform them on images. In *9th International Conference on Learning Representations, ICLR 2021*, 2021.

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.

Andrew Foong, Wessel Bruinsma, Jonathan Gordon, Yann Dubois, James Requeima, and Richard Turner. Meta-learning stationary stochastic process prediction with convolutional neural processes. *Advances in Neural Information Processing Systems*, 33:8284–8295, 2020.

Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Rezende, and SM Ali Eslami. Conditional neural processes. In *International Conference on Machine Learning*, pp. 1704–1713. PMLR, 2018a.

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J Rezende, SM Eslami, and Yee Whye Teh. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018b.

Jonathan Gordon, Wessel P. Bruinsma, Andrew Y. K. Foong, James Requeima, Yann Dubois, and Richard E. Turner. Convolutional conditional neural processes. In *8th International Conference on Learning Representations, ICLR 2020*.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR 2017*.

Oleg Ivanov, Michael Figurnov, and Dmitry P. Vetrov. Variational autoencoder with arbitrary conditioning. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

Saurav Jha, Dong Gong, Xuesong Wang, Richard E Turner, and Lina Yao. The neural process family: Survey, applications and perspectives. *arXiv preprint arXiv:2209.00517*, 2022.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8110–8119, 2020.

Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. volume 34, 2021.

Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, S. M. Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

Mingyu Kim, Kyeong Ryeol Go, and Se-Young Yun. Neural processes with stochastic attention: Paying more attention to the context dataset. In *10th International Conference on Learning Representations, ICLR 2022*, 2022.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014*, 2014.

Adam R Kosiorek, Heiko Strathmann, Daniel Zoran, Pol Moreno, Rosalia Schneider, Sona Mokrá, and Danilo Jimenez Rezende. Nerf-vae: A geometry aware 3d scene generative model. In *International Conference on Machine Learning*, pp. 5742–5752. PMLR, 2021.

Jaeho Lee, Jihoon Tack, Namhoon Lee, and Jinwoo Shin. Meta-learning sparse implicit neural representations. volume 34, pp. 11769–11780, 2021.

Juho Lee, Yoonho Lee, Jungtaek Kim, Eunho Yang, Sung Ju Hwang, and Yee Whye Teh. Bootstrapping neural processes. volume 33, pp. 6606–6615, 2020.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pp. 3730–3738, 2015.

Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7210–7219, 2021.

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 *European conference on computer vision*, pp. 405–421. Springer, 2020.

Tung Nguyen and Aditya Grover. Transformer neural processes: Uncertainty-aware meta learning via sequence modeling. In *International Conference on Machine Learning*, pp. 16569–16594. PMLR, 2022.

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.

Michael Oechsle, Lars Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger. Texture fields: Learning texture representations in function space. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4531–4540, 2019.

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.

Sheldon M Ross, John J Kelly, Roger J Sullivan, William James Perry, Donald Mercer, Ruth M Davis, Thomas Dell Washburn, Earl V Sager, Joseph B Boyce, and Vincent L Bristow. *Stochastic processes*, volume 2. Wiley New York, 1996.

Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. In *5th International Conference on Learning Representations, ICLR 2017,*.

Vincent Sitzmann, Eric Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. volume 33, pp. 10136–10147, 2020a.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. volume 33, pp. 7462–7473, 2020b.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. volume 28, 2015.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. *Advances in neural information processing systems*, 29, 2016.

Kenneth O Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic programming and evolvable machines*, 8(2):131–162, 2007.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. volume 33, pp. 7537–7547, 2020.

Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. volume 33, pp. 19667–19679, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. volume 30, 2017.

Michael Volpp, Fabian Flürenbrock, Lukas Grossberger, Christian Daniel, and Gerhard Neumann. Bayesian context aggregation for neural processes. In *9th International Conference on Learning Representations, ICLR 2021*, 2021.

Qi Wang and Herke Van Hoof. Doubly stochastic variational inference for neural processes with hierarchical latent variables. In *International Conference on Machine Learning*, pp. 10018–10028. PMLR, 2020.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. volume 30, 2017.

APPENDIX A: MORE DETAILS ON MODULATED MLP LAYER

The modulated MLP layer used in our paper is similar to the style modulation in Karras et al. (2020). Mathematically, we denote the weights of a MLP layer (1x1 convolution) by $W \in \mathbb{R}^{d_{in} \times d_{out}}$, where d_{in}, d_{out}, and w_{ij} denote the dimension of input and output, the element in i^{th} row and j^{th} column of W, respectively. We obtain the style vector $\mathbf{s} \in \mathbb{R}^{d_{in}}$ by passing the latent variable $\mathbf{z}$ into two MLP layers. The ith element of style vector s_i is thus used to modulate the parameter of W as,

$$w'_{ij} = s_i \cdot w_{ij}, \quad j = 1, \cdots, d_{out}, \tag{9}$$

where w_{ij} and w'_{ij} denote the original and modulated weights, respectively.

The modulated weights are normalized to preserve training stability,

$$w''_{ij} = w'_{ij} \Big/ \sqrt{\sum_i w'^2_{ij} + \epsilon}, \quad j = 1, \cdots, d_{out}, \tag{10}$$

where ϵ is a very small constant to prevent the denominator to be zero. These two equations describe the mechanism of the modulate MLP used in our method.

APPENDIX B: DETAILED EXPERIMENTAL SETTINGS

Here, we provide the detailed hyper parameters used in all the experiments in Table 6. Among them, some hyper parameters denote to, L_s number of self-attention layers in bottleneck encoder, L_c number of cross-attention layers, K number of hierarchical blocks, d the feature dimension through the entire network. The row of Fourier features describes whether we embed the coordinate into high-frequency embeddings as suggested in (Tancik et al., 2020). β is the hyper parameter used to balance the losses of reconstruction and KL divergence. The reconstruction term describes how we calculate the reconstruction loss $\mathbb{E}_{z \sim q_\phi(z|D_T)}[-\log p_\theta(Y_T|z, X_T, D_C)]$.

	1D synthetic functions	2D images	3D scenes
L_s	3	6	6
L_c	1	4	2
L_K	6	6	4
d	128	512	512
d_z	16	64	64
model size	2.3M	56.7M	34.3M
Fourier features	✗	✓	✓
batch size	100	16	32
iteration number	0.1 million	0.5 million	0.1 million
lr	5e-4	1e-4	1e-4
β	1	0.1	0.001
reconstruction term	Gaussian	discrete logistic mixture (Salimans et al.)	MSE
Training resources	1x V100 16GB	4x V100 16GB (CelebA178)	4x V100 16GB

Table 6: Hyper parameters in our experiments.

In addition, when training for 1D synthetic functions, we will randomly sample 5 to 15 context points and 15 to 25 target points on Matern and RBF kernel.

Training a VNP model for the 1D regression task requires about 5 hours. Training a 2D VNP model on CelebA64 requires about 24 hours. Training a 3D VNP model for novel view synthesis takes around 40 hours. For the inference speed, VNP requires very short time for 1D and 2D tasks. On 1D regression task, VNP takes 0.285 second for testing a batch with batch size of 2000. On the CelebA64 dataset (2D task), our VNP takes around 0.112 second to infer a single batch of size 8. On the 3D Cars dataset, our VNP takes 5.28 second to render an image with a novel view (reasons explained in Appendix C). All these results are tested with a single V100 GPU.

APPENDIX C: COMPARISONS WITH THE OPTIMIZATION-BASED METHOD

In this section, we provide the comparison of our VNP and the previous optimization-based method. We take a representative optimization-based method, SIREN (Sitzmann et al., 2020b) for comparison. We conduct experiments on the 3D Car dataset Chang et al. (2015), and measure both the inference speed and the PSNR of the novel synthesized view. Note that since SIREN requires to optimize the network to fit for a specific signal, the iteration time of SIREN is a part of the inference time. We test the inference speed with a single V100 GPU for both schemes.

The results are shown in Table 7. It is observed that for the task of novel view synthesis conditioned on a single view, our proposed VNP provides much better prediction performance compared with the optimized-based method SIREN, even if SIREN is optimized for many iterations for a given test image. One reason is that VNP can learn the dataset prior, which complements many useful information to predict a specific 3D signal. However, if we optimize the SIREN network to fit the known single view, the network will be initially optimized for the right direction, but will then tend to overfit to this single view after many iterations. As a result, it is observed that when the SIREN network gets better prediction performance as the iteration number increased from 1 to 100. However, the performance decreases if we apply more iterations (such as 300 iterations), compared with that of 100 iterations.

As for the inference speed, we can see that our VNP takes 5.28s to render an image in the novel view with a single forward pass. The reason for such a long inference time is that we use a large number of sampling points in 3D space. If an image is with the resolution of 128×128, to render the RGB value of every pixel, there are $128 \times 128 \times p$ target points, where p is the number of sampled point along each ray (set as $p = 128$ in our experiment). Therefore, processing all these target points with cross-attention requires huge GPU memory. We have to divide the image into several pixel groups, and render these pixel groups sequentially. Currently, we have not implement any optimization of our code. We will improve the implementation of method on tasks involving 3D signals in the future.

Note our VNP only presents slow inference speed on this 3D novel view synthesis task. On the 1D and 2D tasks, since there are not so many target points and the model does not incorporate rendering process, the inference speed of our method is fast.

SIREN						VNP (6 blocks)
Iteration Number	1	10	30	100	300	no iteration
Time (s)	0.21	1.35	3.68	11.59	33.68	5.28
PSNR	11.92	12.27	12.72	12.73	12.00	24.52

Table 7: Novel view synthesis conditioned on a single view. We evaluate the inference time and the prediction performance (PSNR) on ShapeNet Cars (Chang et al., 2015).

APPENDIX D: LIMITATION

The proposed VNP inherits the limitation of NP family. NPs are interesting techniques for meta-learning implicit neural representations due to their reduction of the high cost of training. NPs learns the common knowledge shared by the dataset, enabling fast inference of an unseen signal without the need of finetuning. NPs still cannot work well for dataset including diverse objects (with less shared knowledge), e.g., ImageNet.