NP-PROV: Neural Processes with Position-Relevant-Only Variances

Xuesong Wang[1], Lina Yao[1], Xianzhi Wang[2], Feiping Nie[3], and Boualem Benatallah[1]

[1] University of New South Wales, Sydney 2052, Australia
{xuesong.wang1, lina.yao}@unsw.edu.au
[2] University of Technology Sydney, Sydney 2007, Australia
[3] Northwestern Polytechnical University, Xi'An 710060, China

Abstract. Neural Processes (NPs) families encode distributions over functions to a latent representation given a set of context data, and decode posterior mean and variance at unknown locations. Since mean and variance are derived from the same latent space, they may fail on out-of-domain tasks where fluctuations in function values amplify the model uncertainty. We present a new member named Neural Processes with Position-Relevant-Only Variances (NP-PROV). NP-PROV hypothesizes that a target point close to a context point has small uncertainty, regardless of the function value at that position. The resulting approach derives mean and variance from a function-value-related space and a position-related-only latent space separately. Our evaluation on synthetic and real-world datasets reveals that NP-PROV can achieve state-of-the-art likelihood while retaining a bounded variance when drifts exist in the function value.

Keywords: Uncertainty evaluation · Neural processes · Position-relevant-Only variance.

1 Introduction

Neural networks (NNs) are proven effective in various machine learning tasks for making deterministic predictions. They have the flexibility of describing and fitting any type of data distributions. Nevertheless, most neural networks are incapable of evaluating model stochasticity. They cannot handle tasks where prediction uncertainty is equally crucial, e.g., autonomous driving [17], disease diagnosis [13], and stock market forecasting [3]. In contrast, Gaussian processes (GPs) that model data with an infinite sequence of correlated normal distributions are intrinsically suitable for such problems. Conditioned on some prior knowledge, e.g., context points with positions $\mathbf{X}$ and function values $\mathbf{Y}$, they are able to infer the likelihood of target values $\mathbf{Y}_\star$ at unknown locations $\mathbf{X}_\star$. Despite the advantages, GPs require designing data-specific kernel functions. Also, the cubic computational cost of matrix inversion impedes GPs from handling large-scale data.

The recent progress in models based on Neural Process (NP) [7] has advanced GPs with the fast forward propagation and powerful feature representations of NNs. Basic NPs represent a stochastic process with an encoder-decoder network structure. They encode context data ($\mathbf{X}$,$\mathbf{Y}$) to a latent representation $\mathbf{Z} \sim \mathbf{P}(\mathbf{Z}|\mathbf{X}, \mathbf{Y})$ and decode it to a posterior probability of the target data based on their relationships with the context points $\mathbf{Y} \sim \mathbf{P}(\mathbf{Y}_\star|\mathbf{Z}; \mathbf{X}_\star)$. Recent years have witnessed the success of a family of NP-variants, such as Attentive NP [10], Convolutional Conditional NP [9], and Sequential NP [16]. They improve NP via aggregating context knowledge non-linearly and induce spatial or temporal biases among target points. Nonetheless, they still decode mean and variance from the same latent variable—meaning that their variances are correlated to the context values $\mathbf{Y}$. When there are shifts in the future testing sets(e.g., stock market price with incremental trends), fluctuations in $\mathbf{Y}$ can severely amplify model uncertainty. Therefore, we hypothesize that the precondition of a stabilized model for out-of-domain tasks is that the variance inference being reliant only on locations $\mathbf{X}$ meanwhile unrelated to the function values $\mathbf{Y}$.

In this paper, we introduce a new member named Neural Processes with Position-Relevant-Only Variances (NP-PROV), which derives mean and variance functions from two coupled latent spaces. Mean values are related to context values $\mathbf{Y}$, self-correlation within context locations $\mathbf{X}$, and cross-correlations between context positions $\mathbf{X}$ and target positions $\mathbf{X}_\star$. Variance values exclude function values $\mathbf{Y}$ yet are associated with the self-correlations within the target positions. Our main contributions are as follows:

- We designed an auto-encoder module associated with self-correlations between data points. It reconstructs context data through the module and reduces model uncertainty in high self-correlation scenarios.
- The approach derives mean and variance values from two coupled latent spaces. The position-relevant-only variances prevent the model from estimating oscillating uncertainty when function values are out of the training range.
- The proposed model achieves state-of-the-art performance over on-the-grid and off-the-grid datasets. Specifically, three GP-based kernels and a real-world time series are adopted for off-the-grid tasks. Four image inpainting tasks are evaluated: MNIST, SVHN, celebA and miniImageNet.

2 Background

Gaussian Processes. Let $\mathbf{Y} = [\mathrm{y}_1, ..., \mathrm{y}_N]^\top \in \mathbb{R}^N$ be a set of N observations (as what we call context set here) at input locations $\mathbf{X} = [\mathrm{x}_1, ..., \mathrm{x}_N]^\top \in \mathbb{R}^{N \times D}$, a function $f : \mathbb{R}^D \mapsto \mathbb{R}$, and a likelihood $p(\mathbf{Y}|\mathbf{F}; \mathbf{X})$, where $\mathbf{F} = f(\mathbf{X})$ denotes the function values at the input locations. A Gaussian process (GP) prior is placed on the function f; it models all function values as a jointly Gaussian distribution to infer f. The prior has a mean function $m(\cdot) : \mathbb{R}^D \mapsto \mathbb{R}$ and a covariance function $\mathcal{K}(\cdot, \cdot) : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$. The generative model of the corresponding GP is as follows:

$$p(\mathbf{Y}|\mathbf{F};\mathbf{X}) = \mathcal{N}(m(\mathbf{X}), \mathcal{K}(\mathbf{X},\mathbf{X}^\top)) \tag{1}$$

GP hypothezes similar function values of two inputs that are highly correlated. Given a context set ($\mathbf{X}$, $\mathbf{Y}$), we can perform probabilistic inference and assign posterior distributions over unknown locations $\mathbf{X}_\star = [\mathrm{x}_{\star 1}, ..., \mathrm{x}_{\star M}]^\top$ from the same function. The posterior distribution also follows a joint Gaussian distribution $p(\mathbf{Y}_\star|\mathbf{F};\mathbf{X}_\star,\mathbf{X},\mathbf{Y}) = \mathcal{N}(\mu_\star, \Sigma_\star)$:

$$\begin{aligned} \mu_\star &= m(\mathbf{X}_\star) + \mathbf{K}_\star^\top \mathbf{K}^{-1}(\mathbf{Y} - m(\mathbf{X})) \\ \Sigma_\star &= \mathbf{K}_{\star\star} - \mathbf{K}_\star^\top \mathbf{K}^{-1}\mathbf{K}_\star \end{aligned} \tag{2}$$

where $\mathbf{K} = \mathcal{K}(\mathbf{X},\mathbf{X}^\top), \mathbf{K}_\star = \mathcal{K}(\mathbf{X},\mathbf{X}_\star^\top)$, and $\mathbf{K}_{\star\star} = \mathcal{K}(\mathbf{X}_\star,\mathbf{X}_\star^\top)$. Intuitively, when there is no trend in observations (i.e., $m(\mathbf{X}_\star) = m(\mathbf{X}) = 0$), the mean of $\mathbf{Y}_\star$ is a linear combination of elements in $\mathbf{Y}$. The weights of those elements are associated with the self-correlation of $\mathbf{X}$ (the $\mathbf{K}^{-1}$ part) and cross-correlation between $\mathbf{X}$ and $\mathbf{X}_\star$ (the $\mathbf{K}_\star^\top$ part). The variance equals the total uncertainty of $\mathbf{X}_\star$ (the $\mathbf{K}_{\star\star}$ part) minus the certainty induced from $\mathbf{X}$ (the $\mathbf{K}_\star^\top\mathbf{K}^{-1}\mathbf{K}_\star$ part)and is irrelevant to function values $\mathbf{Y}$.

Convolutional Conditional Neural Processes. As a type of latent model, NPs inherit distribution consistency from GPs, where the context set ($\mathbf{X}$, $\mathbf{Y}$) and the target data set($\mathbf{X}_\star, \mathbf{Y}_\star$) are sampled from the same function and share a latent variable.

Convoluational Conditional Neural Processes (ConvCNP) differ from other NP families in handling testing data outside the range of training data. They introduce a discretization of a continuous range of $\mathbf{X} \cup \mathbf{X}_\star$ named $\mathbf{X}_t$. Convolution operations enable NPs to focus on local receptive fields on the grid and to generalize out-of-range tasks.

The posterior distribution of $p(\mathbf{Y}_\star|\mathbf{F};\mathbf{X}_\star,\mathbf{X}_t,\mathbf{X},\mathbf{Y}) = \mathcal{N}(\mu_\star, \Sigma_\star)$ is also a Gaussian distribution:

$$\begin{aligned} \mu_\star &= \psi_{\star\mu}\{\mathbf{K}_\star^\top \mathbf{CNN}[\psi_t(\mathbf{K}_t^\top \mathbf{Y})]\} \\ \Sigma_\star &= \psi_{\star\Sigma}\{\mathbf{K}_\star^\top \mathbf{CNN}[\psi_t(\mathbf{K}_t^\top \mathbf{Y})]\} \end{aligned} \tag{3}$$

where $\mathbf{K}_\star = \mathcal{K}_\star(\mathbf{X}_\star,\mathbf{X}_t^\top), \mathbf{K}_t = \mathcal{K}_t(\mathbf{X}_t,\mathbf{X}^\top)$ are covariance functions with learnable length scales; $\psi_t, \psi_{\star\mu}$ and $\psi_{\star\Sigma}$ are multi-layer perceptrons (MLPs); the graphical model of ConvCNP is described in Fig 1 (a); $\psi_t(\mathbf{K}_t^\top\mathbf{Y})$ encodes prior knowledge about the function distribution h on the entire grid $\mathbf{X}_t$; $\mathbf{CNN}(\cdot)$ enables model to exhibit translation invariance; $\psi_{\star\mu}(\mathbf{K}_\star^\top\cdot)$ and $\psi_{\star\Sigma}(\mathbf{K}_\star^\top\cdot)$ translate the knowledge to predict target mean and standard deviation for $p(\mathbf{Y}_\star)$.

3 Neural Processes with Position-Relevant-Only Variances

$\mathbf{K}_t$ evaluates correlation between $\mathbf{X}_t$ and $\mathbf{X}$, and $\mathbf{K}_\star$ evaluates cross correlation between $\mathbf{X}_\star$ and $\mathbf{X}_t$. Therefore, $\psi_t(\mathbf{K}_t\mathbf{Y})$ and $\psi_\star(\mathbf{K}_\star^\top\cdot)$ in ConvCNP act as two

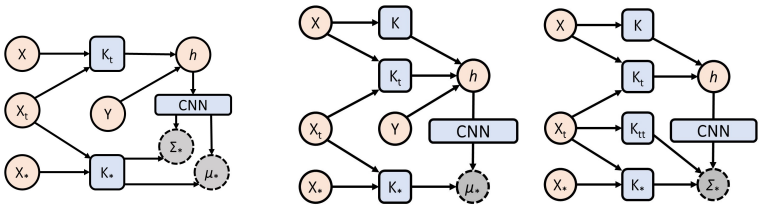

Fig. 1. Graphical Models for (a) ConvCNP (b) NP-PROV mean and (c) NP-PROV variance

cascaded cross-correlation modules in GPs (the $\mathbf{K}_\star^\top$ part). We propose a new member Neural Processes with Position-Relevant-Only Variances (NP-PROV) to improve ConvCNP in two ways:

1. Considering self-correlations within context data. ConvCNP usually considers cross-correlations between target data and context data only. It is possible to further reduce the model uncertainty near the region where context data have high self-correlations.

2. Relying the variance only on the positions $\mathbf{X}$. By removing the original reliance on function values $\mathbf{Y}$ in ConvCNP, we can effectively prevent the model from amplifying uncertainty caused by value fluctuations if it is made to be only associated with relative distances among locations (as in GP).

In the following, an off-the-grid scenario with continuous inputs and an on-the-grid scenario with intrinsically discretized inputs will be detailed.

3.1 Off-the-grid Scenario.

Mean function We follow convCNP to construct the model with two cascaded GP-alike layers. The first layer maps $\mathbf{X}$ to a uniformly discretized grid space $\mathbf{X}_t = [\mathrm{x}_1, ..., \mathrm{x}_t]^\top$ built on the lower and upper range of $\mathbf{X} \cup \mathbf{X}_\star$. The second layer maps the space back to $\mathbf{X}_\star$. As illustrated in Fig 1(b) first line of the workflow, an auto-encoder structure is computed to match $\mathbf{K}^{-1}\mathbf{Y}$ in GPs first. Suppose $\mathbf{K} = \mathcal{K}(\mathbf{X}, \mathbf{X}^\top) \in \mathbb{R}^{N\times N}$ is the self-covariance function; ψ_E and ψ_D are MLPs. Since in GPs, $\mathbf{K}\mathbf{K}^{-1}\mathbf{Y} = \mathbf{Y}$, we hence use an auto-encoder structure to map y_n to a representation $\tilde{h}_n$ and then project it back to y_n. By minimizing the reconstruction loss $\mathcal{L}_2$, we are able to mimic matrix inversion through $\tilde{h}_n$:

$$\begin{aligned} \tilde{h}_n &= \psi_E(\mathbf{K}y_n) \\ \tilde{y}_n = \psi_D(\mathbf{K}\tilde{h}_n), \quad \mathcal{L}_2 &= \frac{1}{N}\sum_{n=1}^{N}(y_n - \tilde{y}_n)^2 \end{aligned} \tag{4}$$

Then, we can compute cross-correlation weights associated with $\mathbf{K}_t = \mathcal{K}(\mathbf{X}_t, \mathbf{X}^\top) \in \mathbb{R}^{T\times N}$ and concatenated with the self-correlation part:

$$\begin{aligned} h'_t &= \mathbf{K}_t[y_n \quad \mathcal{I}] \\ h_t &= \psi_t([\frac{1}{N}\sum_{n=1}^{N}\tilde{h}_n \quad h'_t]) \end{aligned} \tag{5}$$

where h'_t maps the n-th context point to the t-th data in $\mathbf{X}_t$. $\mathcal{I}$ is an additional identity density channel appended to $\mathbf{Y}$ to normalize the scale factor in functional values. A new linear transformation ψ_t fuses averaged self-correlation and cross-correlation information to a latent variable h_t on the t-th data of $\mathbf{X}_t$ as shown the second line of workflow in Fig 1 (b).

Then a CNN takes the obtained condition h_t to suffice translation invariance. It uses a UNet struture to capture global and local patterns. The CNN enables NP-PROV to predict out-of-domain tasks by handling locations outside the training range through filter sliding. When projecting the output back from $\mathbf{X}_t$ to

$\mathbf{X}_\star$, cross-correlation between $\mathbf{X}_\star$ and $\mathbf{X}_t$ is represented as $\mathbf{K}_\star = \mathcal{K}(\mathbf{X}_\star, \mathbf{X}_t^\top) \in \mathbb{R}^{M\times T}$. The overall mean function of the m-th target point in $\mathbf{X}_\star$ is as follows:

$$\mu^\star_m = \psi^\star_\mu(\mathbf{K}_\star \mathbf{CNN}(h_t)) \tag{6}$$

where $\psi_\star$ is the mean MLP transformation, as shown in the third line of the workflow in Fig 1 (b).

Covariance function Based on (2), GP replaces the original values $\mathbf{Y}$ in $\mathbf{K}_\star^\top \mathbf{K}^{-1}\mathbf{Y}$ with the covariance $\mathbf{K}_\star$ in $\mathbf{K}_\star^\top \mathbf{K}^{-1}\mathbf{K}_\star$, and add a module associated with self-correlations within the target data $\mathbf{K}_{\star\star}$. Similarly, as shown in Fig 1 (c), we define $\mathbf{K}_{t'} = \mathcal{K}(\mathbf{X}, \mathbf{X}_t^\top) \in \mathbb{R}^{T\times N}$; $\mathbf{K}_{tt} = \mathcal{K}(\mathbf{X}_t, \mathbf{X}_t^\top) \in \mathbb{R}^{T\times T}$ and two new MLP mappings ψ'_t and ψ_{tt}:

$$y'_n = \psi'_t(\mathcal{I}\mathbf{K}_{t'}), \qquad h_{tt} = \psi_{tt}(\mathcal{I}\mathbf{K}_{tt}) \tag{7}$$

where $\mathcal{I}$ is an identity matrix that helps calculating the sum of the rows of the followed matrix. In this way, y'_n maps $\mathbf{X}_t$ back to $\mathbf{X}$ and replaces values $\mathbf{Y}$ in the mean function. And more importantly, it does not explicitly depend on the function values y. This value-irrelevant input y'_n will generate a new $\tilde{h}_t$ using (5). After the convolution on the new $\tilde{h}_t$, the result is concatenated with h_{tt}. The overall covariance function is as follows:

$$\Sigma^\star_m = \psi^\star_\Sigma(\mathbf{K}_\star[\mathbf{CNN}(\frac{1}{T}\sum_{t=1}^{T} h_{tt}), \quad \tilde{h}_t]) \tag{8}$$

Again, as in Fig1(c), the added elements in NP-PROV can retain reasonable variance regardless of drifts in function values when compared with ConvCNP.

Inference and Learning With $\mu_\star$ and $\Sigma_\star$, the posterior distribution of $\mathbf{Y}_\star$ follows a multivariate normal distribution: $\mathcal{N}(\mu_\star, \Sigma_\star)$. The training objective is to maximize the log-likelihood of target outputs meanwhile minimizing the reconstruction loss on the context set, given all parameters defined as Θ:

$$\theta^\star = \arg\max_{\theta\in\Theta} \sum_{m=1}^{M} \log p(y_m|\mathcal{N}(\mu_\star, \Sigma_\star)) - \frac{1}{N}\sum_{i=1}^{N}(y_i - \tilde{y}_i)^2 \tag{9}$$

3.2 On-the-grid Scenario.

CNN can be applied to On-the-grid data (e.g., images), which are discretized and represent better spatial correlations between context and target data with little efforts. We describe the context inputs $\mathbf{X}$ using a context mask $\mathbf{M}$, where an element value is 1 if a pixel is revealed and 0 otherwise. The target inputs $\mathbf{X}_\star$ formulate the whole image.

Mean function Similar to off-the grid data, self-correlations and cross-correlations weights are computed for the latent variable extraction. Instead of adopting kernel functions, convolutions are implemented on the context masks $\mathbf{M}$ and the context values $\mathbf{M} \odot \mathbf{Y}$:

$$\begin{gathered} \tilde{h}_n = \psi_E(\mathbf{M}), \quad \tilde{M} = \psi_D(\tilde{h}_n), \qquad \mathcal{L}_2 = \| \mathbf{M} - \tilde{\mathbf{M}} \|_2 \\ h_t = \psi_t([\tilde{h}_n, \quad \mathbf{M} \odot \mathbf{Y}]) \end{gathered} \tag{10}$$

where $\mathbf{Y}$ represents a full image, ψ_E, ψ_D, ψ_t are convolutions. ψ_E aims to extract self-correlations within unmasked points. ψ_t shows cross correlations within unmasked pixel values. Then, a UNet CNN structure maps the concatenated latent variable to a translation invariant representation. Finally, a new convolution $\psi_{\star\mu}$ maps the representation to the posterior mean:

$$\mu^\star_m = \psi^\star_\mu(\mathbf{CNN}(h_t)) \tag{11}$$

Covariance function We directly use $\mathbf{M}$ to generate the output-irrelevant latent variable h for variance functions. Also, we add a self-correlation layer on the target masks, i.e., an identity matrix $\mathcal{I}$:

$$\tilde{h}_t = \psi'_t(\mathbf{M}), \quad h_{tt} = \psi_{tt}(\mathcal{I}) \tag{12}$$

The overall function learns a new transformation $\psi^\star_\Sigma$ that maps h and $h_{\star\star}$ to to the posterior covariance. The objective is to maximize a log-likelihood to recover the whole image and to minimize the reconstruction error of the mask simultaneously.

$$\Sigma^\star_m = \psi^\star_\Sigma(\mathbf{CNN}[h_{tt}, \quad \tilde{h}_t]) \tag{13}$$

4 Experiments

We conduct few-shot regression tasks on off-the-grid 1d datasets and on-the-grid images to evaluate the effectiveness of NP-PROV.

4.1 Off-the-grid datasets

We aim to maximize the likelihood of the outputs $\mathbf{Y}_\star$ at unknown locations $\mathbf{X}_\star$, given observations $\mathbf{Y}$ at input locations $\mathbf{X}$. We are interested in the following questions: (a) Are the prediction and uncertainty estimation reasonable? (b) How will the model perform when the testing range of X and Y goes beyond the training data, i.e., out-of-domain tasks? (c) How will the self-correlation affect model prediction? We compare the results of four state-of-the-art neural process members: Neural Process (NP) [7], Conditional Neural Process (CNP) [6], Attentive Neural Process (ANP) [10], and Convolutional Conditional Neural Process(ConvCNP) [9]. We use three challenging kernels to generate synthesised Gaussian processes functions, according to [9]:

- EQ : $\mathcal{K}(x, x') = e^{-\frac{1}{2}(\frac{x-x'}{0.25})^2}$
- Matern $-\frac{5}{2}$: $\mathcal{K}(x, x') = (1 + 4\sqrt{5}d + \frac{5}{3}d^2)e^{-\sqrt{5}d}$ with $d = 4|x - x'|$
- Weakly periodic: $\mathcal{K}(x, x') = e^{-\frac{1}{2}(f_1(x)-f_1(x'))^2-\frac{1}{2}(f_2(x)-f_2(x'))^2} \cdot e^{-\frac{1}{8}(x-x')^2}$, with $f_1(x) = \cos(8\pi x)$ and $f_2(x) = \sin(8\pi x)$

The training data range within $x \in$ [-2, 2] for GP-sampled datasets. Also, a real-world time series dataset Smart Meter [1] is added. It contains energy consumption readings from a sample of 5,567 London Households between November 2011 and February 2014. We select timestamp in days as the input $\mathbf{X}$ and consumption in kWh/ half-hour as the output $\mathbf{Y}$. The x range is set to [0, 2], representing a relative 2-day window from a random clip on the time axis. The number of context points and target points in each task are uniformly distributed in $\mathcal{U}(3, 50)$. A batch of 16 tasks is generated and the total number of tasks in one epoch is 256. We train each model for 200 epochs and then test them using 2,048 new generated tasks for 6 times.

Table 1 shows the log-likelihood (on the probability density function) of five methods. Model performance is presented in Fig 2, which supplement mean and variance results from the original GP as a reference for synthesized datasets. Table 1 shows both ConvCNP and NP-PROV outperform others significantly. The result of NP is quite unstable, due to stochastic encoding of the latent variable. The first column of Fig 2 shows the results with the testing range [-5, 5] for the GP-sampled datasets. NP, CNP, and ANP predict oscillated mean values; therefore, we omit their variance values for display clarity. Both NP-PROV and ConvCNP match well with the original GP. This advantage sources from convolution, where filters can slide along the axis. NP-PROV usually predicts a narrower variance than ConvCNP when the target points are adjacent to context points. This might be attributed to more uncertainty reduction from context self-correlation. When Smart Meter is extended to the interval [-1, 5],

the predicted variance becomes much higher than NP-PROV (as it only adopts cross-correlation), and the target points become far from the context points. NP-PROV mitigates this issue by leveraging the self-correlation on target data.

Table 1. Log-likelihood of off-the-grid datasets (mean ± standard deviation)

Model	EQ	Matern	Weakly Periodic	Smart Meter
NP	-1.20 ± 0.43	-0.82 ± 1.95	-1.75 ± 1.36	0.93 ± 0.23
CNP	-1.10 ± 0.02	-1.36 ± 0.03	-2.04 ± 0.02	1.81 ± 0.06
ANP	-0.35 ± 0.01	-0.75 ± 0.02	-2.09 ± 0.03	1.66 ± 0.05
ConvCNP	2.15 ± 0.05	0.83 ± 0.06	-1.15 ± 0.01	**2.65 ± 0.06**
NP-PROV	**2.20 ± 0.02**	**0.90 ± 0.03**	**-1.00 ± 0.02**	2.32 ± 0.05
ConvCNP (self)	**77.60 ± 2.33**	40.26 ± 0.81	-47.63 ± 0.75	0.95 ± 0.00
NP-PROV (self)	77.55 ± 2.61	**44.14 ± 1.03**	**-40.96 ± 0.89**	**0.99 ± 0.01**

We analyze the impact of self-correlation on the model. In the second column of Fig 2, we give context data in smaller intervals of [-1, -0.5] ∪ [1.2, 1.7] for GP-sampled datasets and [0.2, 0.4] ∪ [0.8, 1.1] for Smart Meter to get higher self-correlations between context points. Also, in Table 1, we present the log-likelihood of ConvCNP (self) and NP-PROV (self) where target points are adjacent to these compacted regions of context data. NP-PROV predicted lower variances on target points near compacted regions, indicating the ability to capture the self-correlation between context points.

Finally, we modify the testing GP generator to $y = 10 \times \mathcal{GP}(x)$ and 1.5 times of the original Smart Meter to evaluate model performance when testing $\mathbf{Y}$ goes beyond the training range. The third column of Fig 2 reveals that all methods are too conservative in adaptation due to the normalization effect of activation layers. However, the variance of NP-PROV on the context data is close to zero, and it is not affected by the explosion of target values, which suffice the hypothesis of a stochastic process. Such an advantage is crucial when there are shifts in time series data.

4.2 On-the-grid datasets

Given different proportions of pixel values, the objective is to complete the whole image and estimate uncertainty at unknown pixels. Four image datasets are introduced: MNIST, SVHN, celebA 32×32 and miniImageNet [5]. Except MNIST, all the other datasets are RBG images. MiniImageNet is used to verify the model capability of recovering images of unseen classes. The context points are sampled from $\mathcal{U}(\frac{n_{total}}{100}, \frac{n_{total}}{2})$ and the target points are n_{total}, i.e., the whole image.

Table 2 demonstrates the log-likelihood of the comparing methods on testing sets. The results of miniImageNet are only displayed for ConvCNP and NP-PROV since other baselines are overwhelmed by the image size 84×84. Fig 3

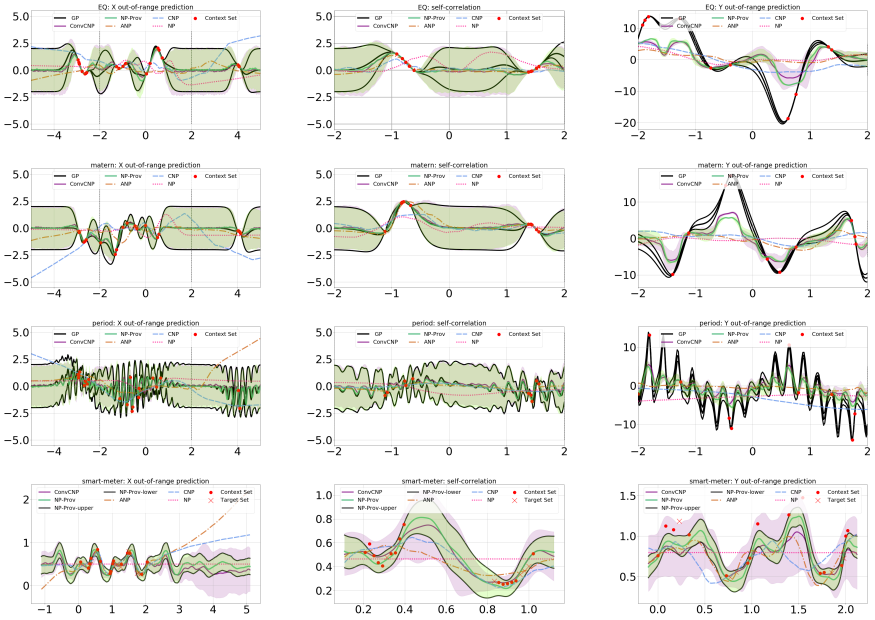

Fig. 2. Model predictions on 4 datasets: EQ (the first row), Matern (the second row), Weakly period (the third row) and Smart meter (the fourth row). The first column is out-of-x-range prediction, the second column is self-correlation prediction, the third column is out-of-y-range prediction.

shows the prediction results on the datasets. NP-PROV and miniImageNet massively outperform other baselines. NP-PROV achieves better results on SVHN and miniImageNet. From the variance results in Fig 3, most of the methods don't exhibit explainable variances in these two methods, because they are similar to "unsupervised" scenarios where one or a few images represent a new class, hence the methods are unable to gain enough variance information from the pixel values. Whereas there are only limited patterns for a digit or an outline of a face in MNIST and celebA, other baselines adopting pixel values can learn a stabilized mean and display variance on the edges.

Table 2. Log-likelihood of on-the-grid datasets (mean ± standard deviation)

Model	MNIST	SVHN	celebA 32×32	miniImageNet
NP	0.65 ± 4e-4	3.21 ± 6e-4	2.78 ± 1e-3	N/A
CNP	1.94 ± 4e-2	4.48 ± 5e-3	3.09 ± 4e-2	N/A
ANP	0.95 ± 3e-3	3.50 ± 5e-3	2.32 ± 4e-2	N/A
ConvCNP	**2.98 ± 4e-2**	6.03 ± 2e-1	**6.35 ± 2e-1**	3.65 ± 4e-2
NP-PROV	2.66 ± 3e-2	**8.24 ± 5e-2**	5.11 ± 1e-2	**4.39 ± 2e-1**

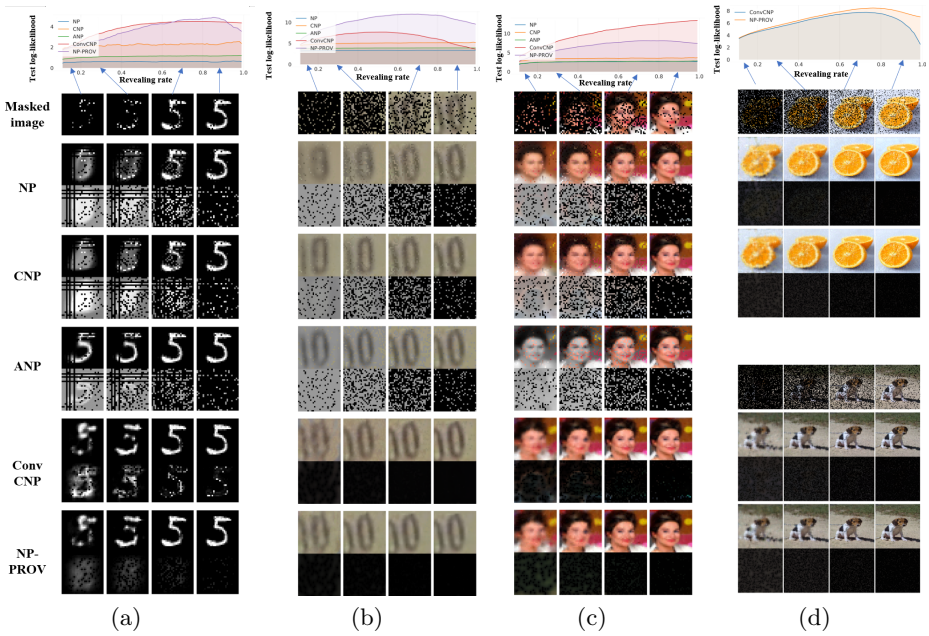

Fig. 3. Test log-likelihood w.r.t revealing rates (the upper row). The lower row presents some samples from (a) MNIST, (b) SVHN, and (c) celebA $\times 32$ and miniImageNet (d) with different revealing rates: 10%, 30%, 70%, 90%. The first row in every method predicts mean and the second row predicts variance.

5 Related Work

Since Gaussian processes suffer high computational cost on matrix inverse, recent work focuses on adopting fast forward-feeding neural networks to substitute original GP. A group of Neural Processes-based models is proposed based on variational inference and ELBO (Evidence Lower BOund) [15]. Neural process families abstract a latent variable or a function from context data and decode the function to the target data based on the relationships between target data and context data. The variations of NPs mostly depend on how the relationships are presented. The original NP [7] and conditional NP [6] adopt mean function to extract stochastic and deterministic latent variables that suffice the exchangeability of a stochastic process. Attentive NP [10] considers the self-attention between context points and cross-attention between context points and target points so that attentive aggregation towards latent variables can be used. Regarding nonstructural relationships between data points, Sequential NP [16] and recurrent NP [18] address the temporal correlations in time series. Convolutional conditional NP [9] utilizes the translation equivariance of convolution to predict on out-of-training range target locations. Functional NP [14]

and Graph NP [2] exploit topological relationships between context and target nodes. Similar to Meta-Agnostic-Meta-Learning(MAML), residual NP [11] assumes that few shot tasks share a unified latent variable with minor variations on each task. Thus, the latent variable of the context data is adapted based on the prediction residues. Besides, numerous work focuses on enhancing Gaussian Processes with non-linear feature extraction using neural networks. NNGP [12] and ConvNet [8] explore the relationships between Gaussian processes and one layer neural networks from the perspective of Bayesian inference and approximation. Deep Gaussian processes [4] and deep kernel learning [19] substitute manual designed kernels with neural networks to extract higher-dimensional features.

6 Conclusions and Discussions

We introduced NP-PROV, a new member of Neural Processes that derive variances from a position-relevant-only latent space. We verify that NP-PROV can estimate bounded uncertainty when context data have high self-correlations or function values are out-of-the-training range. We mitigate the fluctuations in prediction variances when there exists a shift in function values. We believe that NP-PROV opens a door of rethinking the relationships between the mean and variance in Neural Processes. This work leaves multiple avenues for future improvements. It would be interesting to see the mean derivation be more adaptive to out-of-the training range. Also, unifying on-and-off-the-grid version of NP-PROV to fit in higher-dimensional space.

References

1. 3springs: Neural processes for sequential data (2019), available at: https://github.com/3springs/attentive-neural-processes
2. Carr, A.N., Wingate, D.: Graph neural processes: Towards bayesian graph neural networks. arXiv preprint arXiv:1902.10042 (2019)
3. Christou, C., Cunado, J., Gupta, R., Hassapis, C.: Economic policy uncertainty and stock market returns in pacificrim countries: Evidence based on a bayesian panel var model. Journal of Multinational Financial Management **40**, 92–102 (2017)
4. Damianou, A., Lawrence, N.: Deep gaussian processes. In: Artificial Intelligence and Statistics. pp. 207–215 (2013)
5. Deleu, T., Würfl, T., Samiei, M., Cohen, J.P., Bengio, Y.: Torchmeta: A Meta-Learning library for PyTorch (2019), https://arxiv.org/abs/1909.06576, available at: https://github.com/tristandeleu/pytorch-meta
6. Garnelo, M., Rosenbaum, D., Maddison, C.J., Ramalho, T., Saxton, D., Shanahan, M., Teh, Y.W., Rezende, D.J., Eslami, S.: Conditional neural processes. arXiv preprint arXiv:1807.01613 (2018)
7. Garnelo, M., Schwarz, J., Rosenbaum, D., Viola, F., Rezende, D.J., Eslami, S., Teh, Y.W.: Neural processes. arXiv preprint arXiv:1807.01622 (2018)
8. Garriga-Alonso, A., Rasmussen, C.E., Aitchison, L.: Deep convolutional networks as shallow gaussian processes. arXiv preprint arXiv:1808.05587 (2018)

9. Gordon, J., Bruinsma, W.P., Foong, A.Y., Requeima, J., Dubois, Y., Turner, R.E.: Convolutional conditional neural processes. arXiv preprint arXiv:1910.13556 (2019)
10. Kim, H., Mnih, A., Schwarz, J., Garnelo, M., Eslami, A., Rosenbaum, D., Vinyals, O., Teh, Y.W.: Attentive neural processes. arXiv preprint arXiv:1901.05761 (2019)
11. Lee, B.J., Hong, S., Kim, K.: Residual neural processes. In: AAAI 2020 (2020)
12. Lee, J., Bahri, Y., Novak, R., Schoenholz, S.S., Pennington, J., Sohl-Dickstein, J.: Deep neural networks as gaussian processes. arXiv preprint arXiv:1711.00165 (2017)
13. Lorenzi, M., Filippone, M., Frisoni, G.B., Alexander, D.C., Ourselin, S., Initiative, A.D.N., et al.: Probabilistic disease progression modeling to characterize diagnostic uncertainty: application to staging and prediction in alzheimer's disease. NeuroImage **190**, 56–68 (2019)
14. Louizos, C., Shi, X., Schutte, K., Welling, M.: The functional neural process. In: Advances in Neural Information Processing Systems. pp. 8743–8754 (2019)
15. Rudner, T.G., Fortuin, V., Teh, Y.W., Gal, Y.: On the connection between neural processes and gaussian processes with deep kernels. In: Workshop on Bayesian Deep Learning, NeurIPS (2018)
16. Singh, G., Yoon, J., Son, Y., Ahn, S.: Sequential neural processes. In: Advances in Neural Information Processing Systems. pp. 10254–10264 (2019)
17. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NIPS (2017)
18. Wei, J., Dolan, J.M., Snider, J.M., Litkouhi, B.: A point-based mdp for robust single-lane autonomous driving behavior under uncertainties. In: 2011 IEEE International Conference on Robotics and Automation. pp. 2586–2592 (2011)
19. Willi, T., Masci, J., Schmidhuber, J., Osendorfer, C.: Recurrent neural processes. arXiv preprint arXiv:1906.05915 (2019)
20. Wilson, A.G., Hu, Z., Salakhutdinov, R., Xing, E.P.: Deep kernel learning. In: Artificial Intelligence and Statistics. pp. 370–378 (2016)

A Compuational Complexity

An attention layer takes $\mathcal{O}(nmd)$, where n and m are the key and query lengths, and d implies the weight dimension [**?**]. A convolutional layer costs $\mathcal{O}(nkdf)$, where k and f refer to the kernel size and channel depth. Therefore, two MLPs in encoder and decoder cost $\mathcal{O}(msd)$ and $\mathcal{O}(nsd)$, where m and n refer to the context and target set sizes, and s refers to the grid length of the discretization ($s \gg m, n$). The computationally expensive part CNN costs $\mathcal{O}(skdf)$ for convolutions on the $\mathcal{S}$ space. Compared with ConvCNP, the latent path in NP-PROV brought in multiple MLPs that costs $\mathcal{O}(s)$. By tunning the output dimensions of MLPs we can preserve the total number of parameters in convolution and are thus affordable.

B Off-the-grid Datasets Experiments

Datasets Details All models were trained for 200 epochs for GP-based synthetic datasets. Each epoch contains 256 batches with the batch size being 16. After

each epoch, a new batch of 60 tasks were validated and the number of testing tasks was 2048. The Smart Meter dataset is split into 1,710 training batches, 484 validating batches, and 239 testing batches ($7:2:1$) of batch size 16, with a sequence of 100 datapoints per batch. As the data is collected half-hourly, one batch include approximately four days' data. The numbers of context data and of target data are sampled from $\mathcal{U}(3, 50)$ for all the datasets.

Model Structures and Training Details We present model structures in Fig. 4, where Fig. 4(a–b) show how mean and variance are derived from related variables and Table 3 explains every linear transformation in details. The architecture of CNN is detailed in Fig. 4(c). The stride for each convolution is 2. The blue blocks represent convolution layers and the orange ones denote transposed convolution layers. Skipped concatenations are adopted on layers of the same channel depth. During the training process, we use a learning rate of 3e-4 for all the GP-based datasets and 10^{-3} for the Smart Meter. Weight decay of 10^{-5} is applied to all model parameters.

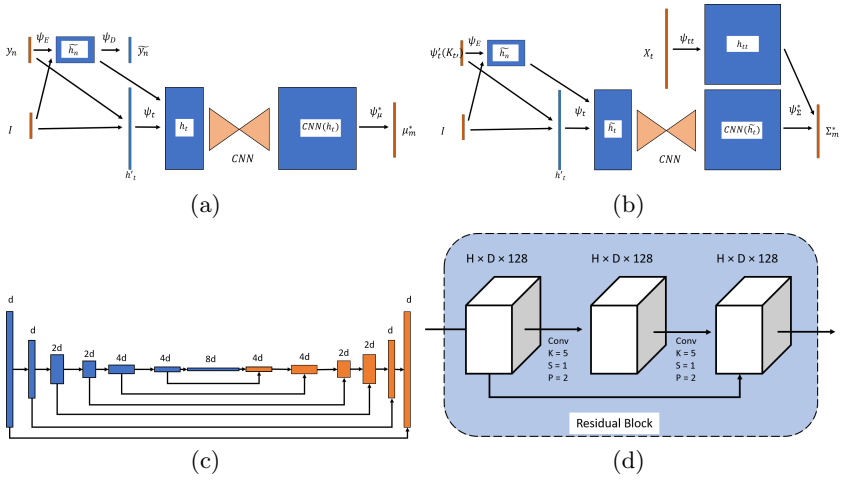

Fig. 4. Model structures on (a) mean derivations, (b)variance derivations, (c) UNet (d) Residual Structure in CNN

C On-the-grid Datasets Experiments

Dataset Details We use MNIST, SVHN, celebA, and miniImageNet datasets in our experiments. The image sizes and the numbers of training and testing samples are presented in Table 4. We split the original training datasets into 7:3 training sets and validating sets for model selection. For RGB images, we scale the pixel values to [0, 1] by dividing them by 255. For miniImageNet dataset, we

Table 3. Notations for off-the-grid datasets

Module	Off-the-grid	On-the-grid	Notation
ψ_E	Linear (2, 8)	Conv2d (c, 128, 9, 1)	encoder for mimicking matrix inversion
ψ_D	Linear (8, 1)	Conv2d (128, c, 9, 1)	decoder for mimicking matrix inversion
ψ_t	Linear (10, 8)	Conv2d (c, 127, 9, 1)	context to grid mapping
$\psi_\mu^\star$	Linear (16, 1)	Conv2d (c, 128, 9, 1)	grid to target mean mapping
$\psi_{t'}$	Linear (2, 1)	Conv2d (128, c, 1, 1)	grid to context mapping
ψ_{tt}	Linear (2, 16)	Conv2d (c, 128, 9, 1)	grid self-correlation
$\psi_\Sigma^\star$	Linear (32, 1)	Conv2d (256, c, 1, 1)	grid to target variance mapping

use 4 samples per batch with 5-way-5-shot images and train the model on 200 batches, resulting in overall $4 \times 5 \times 5 \times 200 = 20,000$ images. The testing set was set to 1-way-1-shot of 63 samples. The batch size for all the other datasets is 16.

Table 4. Data descriptions for on-the-grid datasets

Dataset	Image Size	Training Size	Validating Size	Testing Size
MNIST	(28, 28, 1)	42,000	18,000	10,000
SVHN	(28, 28, 3)	51,279	21,978	26,032
celebA	(32, 32, 3)	141,819	60780	2,592
miniImageNet	(84, 84, 3)	20,000	N/A	63

Model Structures and Training Details The linear transformations (used for on-the-grid datasets) were replaced by convolution operations. The weights and filter sizes are given in Table 3. The padding value is 2 for all the filters, which keeps the output size equalling the input size. The inner CNN structure cascaded four blocks of residual networks detailed in Fig 4 (d). The learning rate was set to be 5e-4 for all the datasets. The training epoch is 100 with early stopping set to 15 epochs.