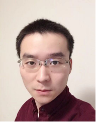
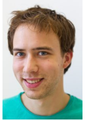
Doubly Stochastic Variational Inference for Neural Processes with Hierarchical Latent Variables

Q. Wang & Herke van Hoof





Amsterdam Machine Learning Lab

ICML 2020

Universiteit van Amsterdam

AMLAB Amsterdam Machine Learning Lab

Highlights in this Work

Highlights in this Work

- A systematical revisit to $\mathcal{SP}$s with an Implicit Latent Variable Model
  - conceptualization of latent $\mathcal{SP}$ models
  - comprehension about $\mathcal{SP}$s with LVMs

Highlights in this Work

- A systematical revisit to $\mathcal{SP}$s with an Implicit Latent Variable Model
  - conceptualization of latent $\mathcal{SP}$ models
  - comprehension about $\mathcal{SP}$s with LVMs
- A novel exchangeable $\mathcal{SP}$ within a Hierarchical Bayesian Framework
  - formalization of a hierarchical $\mathcal{SP}$
  - plausible approximate inference method

Highlights in this Work

- A systematical revisit to $\mathcal{SP}$s with an Implicit Latent Variable Model
  - conceptualization of latent $\mathcal{SP}$ models
  - comprehension about $\mathcal{SP}$s with LVMs
- A novel exchangeable $\mathcal{SP}$ within a Hierarchical Bayesian Framework
  - formalization of a hierarchical $\mathcal{SP}$
  - plausible approximate inference method
- Competitive performance on extensive Uncertainty-aware Applications
  - high dimensional regressions on simulators/real-world dataset
  - classification and o.o.d. detection on image dataset

Outline of this Talk

1. Motivation for $\mathcal{SP}$s
2. Study of $\mathcal{SP}$s with LVMs
3. NP with Hierarchical Latent Variables
4. Experiments and Applications

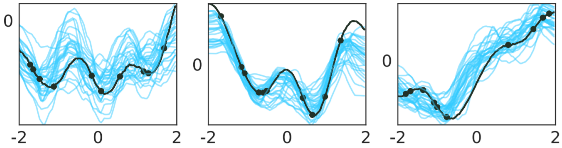
Motivation for $\mathcal{SP}$s

Why Do We Need *Stochastic Processes*?

The stochastic process ($\mathcal{SP}$) is a math tool to describe the distribution over functions. (Fig. refers to [1])

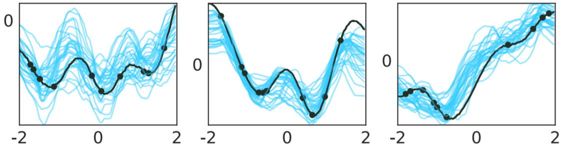
Why Do We Need *Stochastic Processes*?

The stochastic process ($\mathcal{SP}$) is a math tool to describe the distribution over functions. (Fig. refers to [1])



Why Do We Need *Stochastic Processes*?

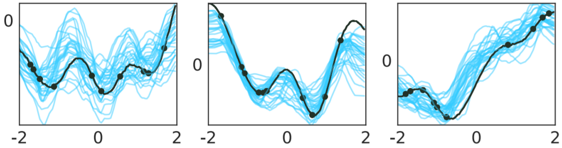
The stochastic process ($\mathcal{SP}$) is a math tool to describe the distribution over functions. (Fig. refers to [1])



- Flexible to handle correlations among samples : significant for non-*i.i.d.* dataset ;

Why Do We Need *Stochastic Processes*?

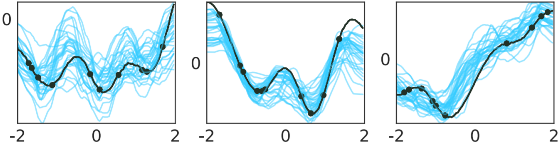
The stochastic process ($\mathcal{SP}$) is a math tool to describe the distribution over functions. (Fig. refers to [1])



- Flexible to handle correlations among samples : significant for non-*i.i.d.* dataset ;
- Quantify uncertainty in risk-sensitive applications : *e.g.* forecast $p(s_{t+1}|s_t, a_t)$ in autonomous driving [2] ;

Why Do We Need *Stochastic Processes*?

The stochastic process ($\mathcal{SP}$) is a math tool to describe the distribution over functions. (Fig. refers to [1])



- Flexible to handle correlations among samples : significant for non-*i.i.d.* dataset ;
- Quantify uncertainty in risk-sensitive applications : *e.g.* forecast $p(s_{t+1}|s_t, a_t)$ in autonomous driving [2] ;
- Model distributions instead of point estimates : working as a *generative model* for more realizations [3].

Two Consistencies in Exchangeable $\mathcal{SP}$s

Some required properties for *exchangeable stochastic process* ρ [4] :

Two Consistencies in Exchangeable $\mathcal{SP}$s

Some required properties for *exchangeable stochastic process* ρ [4] :

- **Marginalization Consistency.** For any finite collection of random variables $\{y_1, y_2, \ldots, y_{N+M}\}$, the probability after *marginalization* over subset is unchanged.

$$\int \rho_{x_{1:N+M}}(y_{1:N+M}) dy_{N+1:N+M} = \rho_{x_{1:N}}(y_{1:N}) \tag{1.1}$$

- **Exchangeability Consistency.** Any random *permutation* over set of variables does not influence joint probability.

$$\rho_{x_{1:N}}(y_{1:N}) = \rho_{x_{\pi(1:N)}}(y_{\pi(1:N)}) \tag{1.2}$$

Two Consistencies in Exchangeable $\mathcal{SP}$s

Some required properties for *exchangeable stochastic process* ρ [4] :

- **Marginalization Consistency.** For any finite collection of random variables $\{y_1, y_2, \ldots, y_{N+M}\}$, the probability after *marginalization* over subset is unchanged.

$$\int \rho_{x_{1:N+M}}(y_{1:N+M}) dy_{N+1:N+M} = \rho_{x_{1:N}}(y_{1:N}) \tag{1.1}$$

- **Exchangeability Consistency.** Any random *permutation* over set of variables does not influence joint probability.

$$\rho_{x_{1:N}}(y_{1:N}) = \rho_{x_{\pi(1:N)}}(y_{\pi(1:N)}) \tag{1.2}$$

With these two conditions, an exchangeable $\mathcal{SP}$ can be induced. (Refer to *Kolmogorov Extension Theorem*)

$\mathcal{SP}$s in Progress and Primary Concerns

Crucial properties for $\mathcal{SP}$s :

- Scalability in large-scale dataset:
- Flexibility in distributions:
- Extension to high dimensions:

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
- Neural Processes (NPs)

$\mathcal{SP}$s in Progress and Primary Concerns

Crucial properties for $\mathcal{SP}$s :

- Scalability in large-scale dataset: $\rightarrow$ Optimization/Computational bottleneck
- Flexibility in distributions:
- Extension to high dimensions:

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
- Neural Processes (NPs)

$\mathcal{SP}$s in Progress and Primary Concerns

Crucial properties for $\mathcal{SP}$s :

- Scalability in large-scale dataset:
  $\rightarrow$ Optimization/Computational bottleneck
- Flexibility in distributions:
  $\rightarrow$ Non-Gaussian or Multi-modal property
- Extension to high dimensions:

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
- Neural Processes (NPs)

$\mathcal{SP}$s in Progress and Primary Concerns

Crucial properties for $\mathcal{SP}$s :

- Scalability in large-scale dataset:
  $\rightarrow$ Optimization/Computational bottleneck
- Flexibility in distributions:
  $\rightarrow$ Non-Gaussian or Multi-modal property
- Extension to high dimensions:
  $\rightarrow$ Correlations among or across Input/Output

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
- Neural Processes (NPs)

$\mathcal{SP}$s in Progress and Primary Concerns

Crucial properties for $\mathcal{SP}$s :

- Scalability in large-scale dataset:
  $\rightarrow$ Optimization/Computational bottleneck
- Flexibility in distributions:
  $\rightarrow$ Non-Gaussian or Multi-modal property
- Extension to high dimensions:
  $\rightarrow$ Correlations among or across Input/Output

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
  $\rightarrow$ less scalable with computational complexity $\mathcal{O}(N^3)$
- Neural Processes (NPs)

$\mathcal{SP}$s in Progress and Primary Concerns

Crucial properties for $\mathcal{SP}$s :

- Scalability in large-scale dataset:
  $\rightarrow$ Optimization/Computational bottleneck
- Flexibility in distributions:
  $\rightarrow$ Non-Gaussian or Multi-modal property
- Extension to high dimensions:
  $\rightarrow$ Correlations among or across Input/Output

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
  $\rightarrow$ less scalable with computational complexity $\mathcal{O}(N^3)$
  $\rightarrow$ less flexible with Gaussian distributions
- Neural Processes (NPs)

$\mathcal{SP}$s in Progress and Primary Concerns

Crucial properties for $\mathcal{SP}$s :

- Scalability in large-scale dataset:
  $\rightarrow$ Optimization/Computational bottleneck
- Flexibility in distributions:
  $\rightarrow$ Non-Gaussian or Multi-modal property
- Extension to high dimensions:
  $\rightarrow$ Correlations among or across Input/Output

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
  $\rightarrow$ less scalable with computational complexity $\mathcal{O}(N^3)$
  $\rightarrow$ less flexible with Gaussian distributions
- Neural Processes (NPs)
  $\rightarrow$ more scalable with computational complexity $\mathcal{O}(N)$

$\mathcal{SP}$s in Progress and Primary Concerns

Crucial properties for $\mathcal{SP}$s :

- Scalability in large-scale dataset:
  $\rightarrow$ Optimization/Computational bottleneck
- Flexibility in distributions:
  $\rightarrow$ Non-Gaussian or Multi-modal property
- Extension to high dimensions:
  $\rightarrow$ Correlations among or across Input/Output

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
  $\rightarrow$ less scalable with computational complexity $\mathcal{O}(N^3)$
  $\rightarrow$ less flexible with Gaussian distributions
- Neural Processes (NPs)
  $\rightarrow$ more scalable with computational complexity $\mathcal{O}(N)$
  $\rightarrow$ more flexible with no explicit distributions

Study of $\mathcal{SP}$s with LVMs

Deep Latent Variable Model as $\mathcal{SP}$s

Here we present an implicit Latent Variable Model for $\mathcal{SP}$s :

- Generation paradigm with (potentially correlated) latent variables :

- Predictive distribution in $\mathcal{SP}$s : Let the *context* and *target input* be $\mathcal{C} = \{(x_i, y_i) | i = 1, 2, \ldots, N\}$ and x_T, the computation is

(2.3)

mostly intractable.

Deep Latent Variable Model as $\mathcal{SP}$s

Here we present an implicit Latent Variable Model for $\mathcal{SP}$s :

- Generation paradigm with (potentially correlated) latent variables :

$$\underbrace{z_i}_{\text{index depend. l.v.}} = \underbrace{\phi(x_i)}_{\text{deter. term}} + \underbrace{\epsilon(x_i)}_{\text{stoch. term}} \tag{2.1}$$

- Predictive distribution in $\mathcal{SP}$s : Let the *context* and *target input* be $\mathcal{C} = \{(x_i, y_i) | i = 1, 2, \ldots, N\}$ and x_T, the computation is

(2.3)

mostly intractable.

Deep Latent Variable Model as $\mathcal{SP}$s

Here we present an implicit Latent Variable Model for $\mathcal{SP}$s :

- Generation paradigm with (potentially correlated) latent variables :

$$\underbrace{z_i}_{\text{index depend. l.v.}} = \underbrace{\phi(x_i)}_{\text{deter. term}} + \underbrace{\epsilon(x_i)}_{\text{stoch. term}} \tag{2.1}$$

$$\underbrace{y_i}_{obs.} = \underbrace{\varphi(x_i, z_i)}_{\text{trans.}} + \underbrace{\zeta_i}_{\text{obs. noise}} \tag{2.2}$$

- Predictive distribution in $\mathcal{SP}$s : Let the *context* and *target input* be $\mathcal{C} = \{(x_i, y_i) | i = 1, 2, \dots, N\}$ and x_T, the computation is

(2.3)

mostly intractable.

Deep Latent Variable Model as $\mathcal{SP}$s

Here we present an implicit Latent Variable Model for $\mathcal{SP}$s :

- Generation paradigm with (potentially correlated) latent variables :

$$\underbrace{z_i}_{\text{index depend. l.v.}} = \underbrace{\phi(x_i)}_{\text{deter. term}} + \underbrace{\epsilon(x_i)}_{\text{stoch. term}} \tag{2.1}$$

$$\underbrace{y_i}_{obs.} = \underbrace{\varphi(x_i, z_i)}_{\text{trans.}} + \underbrace{\zeta_i}_{\text{obs. noise}} \tag{2.2}$$

- Predictive distribution in $\mathcal{SP}$s : Let the *context* and *target input* be $\mathcal{C} = \{(x_i, y_i) | i = 1, 2, \dots, N\}$ and x_T, the computation is

$$p_\theta(z_T | x_C, y_C, x_T) = \frac{p(z_C, z_T)}{\int p(z_C, z_T) dz_C}, \tag{2.3}$$

mostly intractable.

Deep Latent Variable Model as $\mathcal{SP}$s

Here we present an implicit Latent Variable Model for $\mathcal{SP}$s :

- Generation paradigm with (potentially correlated) latent variables :

$$\underbrace{z_i}_{\text{index depend. l.v.}} = \underbrace{\phi(x_i)}_{\text{deter. term}} + \underbrace{\epsilon(x_i)}_{\text{stoch. term}} \tag{2.1}$$

$$\underbrace{y_i}_{obs.} = \underbrace{\varphi(x_i, z_i)}_{\text{trans.}} + \underbrace{\zeta_i}_{\text{obs. noise}} \tag{2.2}$$

- Predictive distribution in $\mathcal{SP}$s : Let the *context* and *target input* be $\mathcal{C} = \{(x_i, y_i) | i = 1, 2, \ldots, N\}$ and x_T, the computation is

$$p_\theta(z_T | x_C, y_C, x_T) = \frac{p(z_C, z_T)}{\int p(z_C, z_T) dz_C}, \quad y_T \sim p(y_T | x_T, z_T, \zeta) \tag{2.3}$$

mostly intractable.

NP family approximates $\mathcal{SP}$s in the form of LVMs :

- $\mathcal{GP}$ as an exchangeable $\mathcal{SP}$ with latent variables :

- NP as an exchangeable $\mathcal{SP}$ with a global latent variable :

Gaussian Processes & Neural Processes

NP family approximates $\mathcal{SP}$s in the form of LVMs :

- $\mathcal{GP}$ as an exchangeable $\mathcal{SP}$ with latent variables :

$$\rho_x(y) = \int \mathcal{N}(y; z, \tau^{-1}\mathcal{I}) \underbrace{\mathcal{N}(z; m(x), \mathcal{K}(.,.))}_{\text{l.v.}} dz \tag{2.4}$$

- NP as an exchangeable $\mathcal{SP}$ with a global latent variable :

Gaussian Processes & Neural Processes

NP family approximates $\mathcal{SP}$s in the form of LVMs :

- $\mathcal{GP}$ as an exchangeable $\mathcal{SP}$ with latent variables :

$$\rho_x(y) = \int \mathcal{N}(y; z, \tau^{-1}\mathcal{I}) \underbrace{\mathcal{N}(z; m(x), \mathcal{K}(.,.))}_{\text{l.v.}} dz \tag{2.4}$$

- NP as an exchangeable $\mathcal{SP}$ with a global latent variable :

$$\rho_{x_{1:N+M}}(y_{1:N+M}) = \int \prod_{i=1}^{N+M} \underbrace{p(y_i|x_i, z_G)}_{\text{trans.}} \underbrace{p(z_G)}_{\text{global l.v.}} dz_G \tag{2.5}$$

NP family approximates $\mathcal{SP}$s in the form of LVMs :

- $\mathcal{GP}$ as an exchangeable $\mathcal{SP}$ with latent variables :

$$\rho_x(y) = \int \mathcal{N}(y; z, \tau^{-1}\mathcal{I}) \underbrace{\mathcal{N}(z; m(x), \mathcal{K}(.,.))}_{\text{l.v.}} dz \tag{2.4}$$

- NP as an exchangeable $\mathcal{SP}$ with a global latent variable :

$$\rho_{x_{1:N+M}}(y_{1:N+M}) = \int \prod_{i=1}^{N+M} \underbrace{p(y_i|x_i, z_G)}_{\text{trans.}} \underbrace{p(z_G)}_{\text{global l.v.}} dz_G \tag{2.5}$$

Remark

Some other models, such as Hierarchical $\mathcal{GP}$s [5] and Deep $\mathcal{GP}$s [6], [7] can also be expressed with LVMs.

Inference for Neural Processes

A general ELBO with **a context prior** in NP models [1] :

Statistics of the context invariant to the order in set instances, such as pooling of element-wise embeddings :

Inference for Neural Processes

A general ELBO with **a context prior** in NP models [1] :

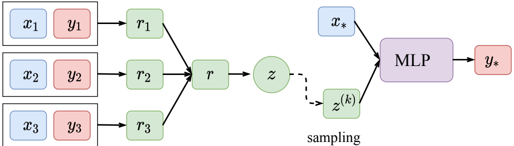
$$\begin{aligned} \ln\left[p(y_T|x_C, y_C, x_T)\right] \geq \mathbb{E}_{q_\phi} \ln\left[\underbrace{p_\theta(y_T|x_T, z_G)}_{\textit{data likelihood}}\right] \\ - D_{KL}\left(\underbrace{q_\phi(z_G|x_C, y_C, x_T, y_T)}_{\textit{global posterior}} \,\|\, \underbrace{p(z_G|x_C, y_C)}_{\textit{global prior}}\right) \end{aligned} \tag{2.6}$$

Statistics of the context invariant to the order in set instances, such as pooling of element-wise embeddings :

A general ELBO with **a context prior** in NP models [1] :

$$\ln \left[p(y_T|x_C, y_C, x_T) \right] \geq \mathbb{E}_{q_\phi} \ln \Big[\underbrace{p_\theta(y_T|x_T, z_G)}_{\textit{data likelihood}} \Big] - D_{KL}\Big(\underbrace{q_\phi(z_G|x_C, y_C, x_T, y_T)}_{\textit{global posterior}} \,\|\, \underbrace{p(z_G|x_C, y_C)}_{\textit{global prior}} \Big) \tag{2.6}$$

Statistics of the context invariant to the order in set instances, such as pooling of element-wise embeddings :



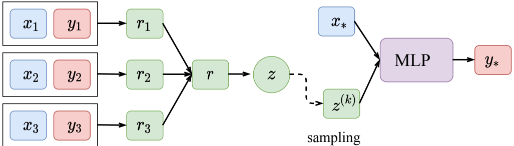
Inference for Neural Processes

A general ELBO with **a context prior** in NP models [1] :

$$\ln\left[p(y_T|x_C, y_C, x_T)\right] \geq \mathbb{E}_{q_\phi} \ln\Big[\underbrace{p_\theta(y_T|x_T, z_G)}_{\textit{data likelihood}}\Big] - D_{KL}\Big(\underbrace{q_\phi(z_G|x_C, y_C, x_T, y_T)}_{\textit{global posterior}} \,\|\, \underbrace{p(z_G|x_C, y_C)}_{\textit{global prior}}\Big) \tag{2.6}$$

Statistics of the context invariant to the order in set instances, such as pooling of element-wise embeddings :

$$r_i = h_\theta(x_i, y_i), \quad r = \bigoplus_{i=1}^{N} r_i, \quad p_\theta(z_C|x_C, y_C) = \mathcal{N}(z_C|[f_\mu(r), f_\sigma(r)]) \tag{2.7}$$



NPs with Hierarchical Latent Variables

Extending NPs from A Hierarchical Bayes Perspective

Our work starts with motivations:

- Hierarchical Bayesian structures $\rightarrow$ more expressiveness.

Extending NPs from A Hierarchical Bayes Perspective

Our work starts with motivations:

- Hierarchical Bayesian structures $\rightarrow$ more expressiveness.
- Involving local l.v. $\rightarrow$ reveal local dependencies across input/output in high-dim cases.

Extending NPs from A Hierarchical Bayes Perspective

Our work starts with motivations:

- Hierarchical Bayesian structures $\rightarrow$ more expressiveness.
- Involving local l.v. $\rightarrow$ reveal local dependencies across input/output in high-dim cases.

As a result, a hierarchical LVM is induced as Doubly Stochastic Variational Neural Process (DSVNP):

Extending NPs from A Hierarchical Bayes Perspective

Our work starts with motivations:

- Hierarchical Bayesian structures $\rightarrow$ more expressiveness.
- Involving local l.v. $\rightarrow$ reveal local dependencies across input/output in high-dim cases.

As a result, a hierarchical LVM is induced as Doubly Stochastic Variational Neural Process (DSVNP):

$$\rho_{x_{1:N+M}}(y_{1:N+M}) = \iint \prod_{i=1}^{N+M} p(y_i|z_G, z_i, x_i) p(z_i|x_i, z_G)p(z_G)dz_{1:N+M}dz_G \tag{3.1}$$

Extending NPs from A Hierarchical Bayes Perspective

Our work starts with motivations:

- Hierarchical Bayesian structures $\rightarrow$ more expressiveness.
- Involving local l.v. $\rightarrow$ reveal local dependencies across input/output in high-dim cases.

As a result, a hierarchical LVM is induced as Doubly Stochastic Variational Neural Process (DSVNP):

$$\rho_{x_{1:N+M}}(y_{1:N+M}) = \iint \prod_{i=1}^{N+M} p(y_i|z_G, z_i, x_i) \\ p(z_i|x_i, z_G)p(z_G)dz_{1:N+M}dz_G \tag{3.1}$$

Remark

DSVNP satisfies **Marginalization** and **Exchangeability Consistencies**, so it is a new exchangeable $\mathcal{SP}$.

Approximate Inference for DSVNP

Exact inference for this hierarchical LVM is mostly intractable, hence approximate inference is used here.

- Evidence Lower Bound for DSVNP :

- Generative (Black Lines) and Recognition Models (Blue/Pink Lines) in Graphs : Specify generative process with black line

Approximate Inference for DSVNP

Exact inference for this hierarchical LVM is mostly intractable, hence approximate inference is used here.

- Evidence Lower Bound for DSVNP :

$$\begin{aligned} \ln\left[p(y_*|x_C, y_C, x_*)\right] \geq \; & \mathbb{E}_{q_{\phi_{1,1}}}\mathbb{E}_{q_{\phi_{2,1}}} \ln[p(y_*|z_G, z_*, x_*)] \\ & -\mathbb{E}_{q_{\phi_{1,1}}}[D_{KL}[q_{\phi_{2,1}}(z_*|z_G, x_*, y_*) \parallel p_{\phi_{2,2}}(z_*|z_G, x_*)]] \\ & -D_{KL}\left[q_{\phi_{1,1}}(z_G|x_C, y_C, x_T, y_T) \parallel p_{\phi_{1,2}}(z_G|x_C, y_C)\right] \end{aligned} \tag{3.2}$$

- Generative (Black Lines) and Recognition Models (Blue/Pink Lines) in Graphs : Specify generative process with black line

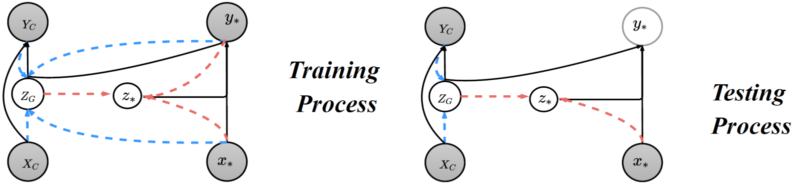
Approximate Inference for DSVNP

Exact inference for this hierarchical LVM is mostly intractable, hence approximate inference is used here.

- Evidence Lower Bound for DSVNP :

$$\begin{aligned} \ln\left[p(y_*|x_C, y_C, x_*)\right] \geq & \mathbb{E}_{q_{\phi_{1,1}}}\mathbb{E}_{q_{\phi_{2,1}}} \ln[p(y_*|z_G, z_*, x_*)] \\ & -\mathbb{E}_{q_{\phi_{1,1}}}[D_{KL}[q_{\phi_{2,1}}(z_*|z_G, x_*, y_*) \parallel p_{\phi_{2,2}}(z_*|z_G, x_*)]] \\ & -D_{KL}\left[q_{\phi_{1,1}}(z_G|x_C, y_C, x_T, y_T) \parallel p_{\phi_{1,2}}(z_G|x_C, y_C)\right] \end{aligned} \quad (3.2)$$

- Generative (Black Lines) and Recognition Models (Blue/Pink Lines) in Graphs : Specify generative process with black line



Training and Testing in Practice

Similar to that in NPs, DSVNP is trained in a SGVB way [8].

- Scalable training with random context points :

- Testing/Forecasting with priors and Monte Carlo estimates :

Training and Testing in Practice

Similar to that in NPs, DSVNP is trained in a SGVB way [8].

- Scalable training with random context points :

Algorithm 1 Variational Inference for DSVNP in Training.

Input: Dataset $\mathcal{D} = \{x_C, y_C; x_T, y_T\}$, Maximum context points N_{max}, etc.
Output: Model parameters ϕ_1, ϕ_2 and θ.
for $i = 1$ **to** m **do**
  Draw some context number $N_C \sim U[1, N_{max}]$;
  Draw mini-batch pair instances $\{(x_C, y_C, x_T, y_T)_{bs}\}_{bs=1}^{B} \sim \mathcal{D}$;
  Feedforward instances to recognition model q_{ϕ_1};
  Feedforward latent variables to generative model p_θ;
  Update parameters by Optimizing Eq. (12):
    $\phi_1 \leftarrow \phi_1 + \alpha \nabla_{\phi_1} \mathcal{L}_{MC}$ ▷ $\phi_1 = [\phi_{1,1}, \phi_{1,2}]$
    $\phi_2 \leftarrow \phi_2 + \alpha \nabla_{\phi_2} \mathcal{L}_{MC}$ ▷ $\phi_1 = [\phi_{2,1}, \phi_{2,2}]$
    $\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}_{MC}$
end for

- Testing/Forecasting with priors and Monte Carlo estimates :

Training and Testing in Practice

Similar to that in NPs, DSVNP is trained in a SGVB way [8].

- Scalable training with random context points :

Algorithm 1 Variational Inference for DSVNP in Training.

Input: Dataset $\mathcal{D} = \{x_C, y_C; x_T, y_T\}$, Maximum context points N_{max}, etc.
Output: Model parameters ϕ_1, ϕ_2 and θ.
for $i = 1$ **to** m **do**
  Draw some context number $N_C \sim U[1, N_{max}]$;
  Draw mini-batch pair instances $\{(x_C, y_C, x_T, y_T)_{bs}\}_{bs=1}^{B} \sim \mathcal{D}$;
  Feedforward instances to recognition model q_{ϕ_1};
  Feedforward latent variables to generative model p_θ;
  Update parameters by Optimizing Eq. (12):
    $\phi_1 \leftarrow \phi_1 + \alpha \nabla_{\phi_1} \mathcal{L}_{MC}$ ▷ $\phi_1 = [\phi_{1,1}, \phi_{1,2}]$
    $\phi_2 \leftarrow \phi_2 + \alpha \nabla_{\phi_2} \mathcal{L}_{MC}$ ▷ $\phi_1 = [\phi_{2,1}, \phi_{2,2}]$
    $\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}_{MC}$
end for

- Testing/Forecasting with priors and Monte Carlo estimates :

$$p(y_*|x_C, y_C, x_*) \approx \frac{1}{KS} \sum_{k=1}^{K} \sum_{s=1}^{S} p_\theta(y_*|x_*, z_*^{(s)}, z_G^{(k)}) \tag{3.3}$$

Training and Testing in Practice

Similar to that in NPs, DSVNP is trained in a SGVB way [8].

- Scalable training with random context points :

Algorithm 1 Variational Inference for DSVNP in Training.

Input: Dataset $\mathcal{D} = \{x_C, y_C; x_T, y_T\}$, Maximum context points N_{max}, etc.
Output: Model parameters ϕ_1, ϕ_2 and θ.
for $i = 1$ **to** m **do**
  Draw some context number $N_C \sim U[1, N_{max}]$;
  Draw mini-batch pair instances $\{(x_C, y_C, x_T, y_T)_{bs}\}_{bs=1}^{B} \sim \mathcal{D}$;
  Feedforward instances to recognition model q_{ϕ_1};
  Feedforward latent variables to generative model p_θ;
  Update parameters by Optimizing Eq. (12):
    $\phi_1 \leftarrow \phi_1 + \alpha \nabla_{\phi_1} \mathcal{L}_{MC}$ ▷ $\phi_1 = [\phi_{1,1}, \phi_{1,2}]$
    $\phi_2 \leftarrow \phi_2 + \alpha \nabla_{\phi_2} \mathcal{L}_{MC}$ ▷ $\phi_1 = [\phi_{2,1}, \phi_{2,2}]$
    $\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}_{MC}$
end for

- Testing/Forecasting with priors and Monte Carlo estimates :

$$p(y_*|x_C, y_C, x_*) \approx \frac{1}{KS} \sum_{k=1}^{K} \sum_{s=1}^{S} p_\theta(y_*|x_*, z_*^{(s)}, z_G^{(k)}) \tag{3.3}$$

using latent variables sampled in prior networks as $z_G^{(k)} \sim p_{\phi_{1,2}}(z_G|x_C, y_C)$ and $z_*^{(s)} \sim p_{\phi_{2,2}}(z_*|z_G^{(k)}, x_*)$.

Experiments and Applications

Toy Experiments

Discoveries in 1-D Simulation Experiments in terms of fitting errors and uncertainty quantification (UQ) :

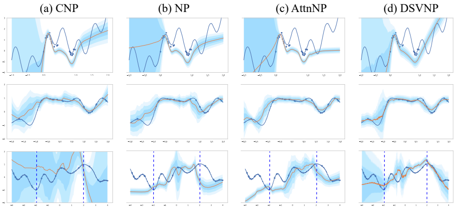
- Episdemic uncertainty in a single curve :

- Interpolation in curves of a $\mathcal{SP}$:

- Extrapolation in curves of a $\mathcal{SP}$:

Toy Experiments

Discoveries in 1-D Simulation Experiments in terms of fitting errors and uncertainty quantification (UQ) :



- Episdemic uncertainty in a single curve : NP/AttnNP $\rightarrow$ over-confident in some regions
- Interpolation in curves of a $\mathcal{SP}$:
- Extrapolation in curves of a $\mathcal{SP}$:

Toy Experiments

Discoveries in 1-D Simulation Experiments in terms of fitting errors and uncertainty quantification (UQ) :

- Episdemic uncertainty in a single curve : NP/AttnNP $\rightarrow$ over-confident in some regions
- Interpolation in curves of a $\mathcal{SP}$: AttnNP $\succ$ DSVNP $\succ$ NP $\succ$ CNP (Fitting/UQ Performance)
- Extrapolation in curves of a $\mathcal{SP}$:

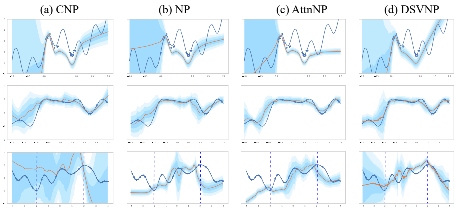


Table 2. Average Negative Log-likelihoods over all target points on realizations from Synthetic Stochastic Process. (Figures in brackets are variances.)

PREDICTION	CNP	NP	ATTNNP	DSVNP
INTER	-0.802 (1E-6)	-0.958 (2E-5)	**-1.149 (8E-6)**	-0.975 (2E-5)
EXTRA	**1.764 (1E-1)**	8.192 (7E1)	8.091 (7E2)	**4.203 (9E0)**

Toy Experiments

Discoveries in 1-D Simulation Experiments in terms of fitting errors and uncertainty quantification (UQ) :

- Episdemic uncertainty in a single curve : NP/AttnNP $\rightarrow$ over-confident in some regions
- Interpolation in curves of a $\mathcal{SP}$: AttnNP $\succ$ DSVNP $\succ$ NP $\succ$ CNP (Fitting/UQ Performance)
- Extrapolation in curves of a $\mathcal{SP}$: Tough for all in fitting; NP/AttnNP $\rightarrow$ over-confident; DSVNP $\rightarrow$ better UQ

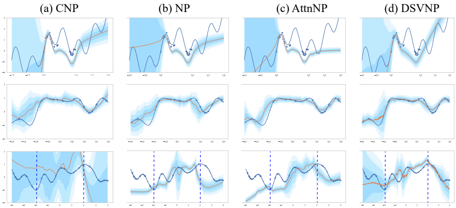


Table 2. Average Negative Log-likelihoods over all target points on realizations from Synthetic Stochastic Process. (Figures in brackets are variances.)

PREDICTION	CNP	NP	ATTNNP	DSVNP
INTER	-0.802 (1E-6)	-0.958 (2E-5)	**-1.149 (8E-6)**	-0.975 (2E-5)
EXTRA	**1.764 (1E-1)**	8.192 (7E1)	8.091 (7E2)	**4.203 (9E0)**

Multi-output Regression: Simulation/Real-world Dataset

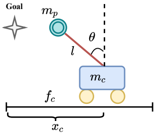
Investigations on (1) system identification on cart-pole transitions [9]; (2) regression on real-world dataset :

- System identification :
- High-dim regression :

Multi-output Regression: Simulation/Real-world Dataset

Investigations on (1) system identification on cart-pole transitions [9]; (2) regression on real-world dataset :

- System identification :



- High-dim regression :

Multi-output Regression: Simulation/Real-world Dataset

Investigations on (1) system identification on cart-pole transitions [9]; (2) regression on real-world dataset :

- System identification :
  MSE & NLL not in accordance; DSVNP & CNP $\rightarrow$ better UQ; DSVNP & AttnNP $\rightarrow$ lower fitting error.
- High-dim regression :

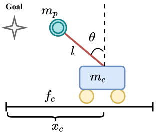


Table 3. Predictive Negative Log-Likelihoods and Mean Square Errors on Cart-Pole State Transition Testing Dataset. (Figures in brackets are variances.)

METRICS	CNP	NP	ATTNNP	DSVNP
NLL	-2.014 (9E-4)	-1.537 (1E-3)	-1.821 (7E-3)	**-2.145 (9E-4)**
MSE	0.096 (3E-4)	0.074 (2E-4)	0.067 (1E-4)	**0.036 (2.1E-5)**

Multi-output Regression: Simulation/Real-world Dataset

Investigations on (1) system identification on cart-pole transitions [9]; (2) regression on real-world dataset :

- System identification :
  MSE & NLL not in accordance; DSVNP & CNP $\rightarrow$ better UQ; DSVNP & AttnNP $\rightarrow$ lower fitting error.
- High-dim regression :
  Hierarchical latent variables advance performance significantly.

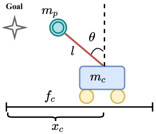


Table 3. Predictive Negative Log-Likelihoods and Mean Square Errors on Cart-Pole State Transition Testing Dataset. (Figures in brackets are variances.)

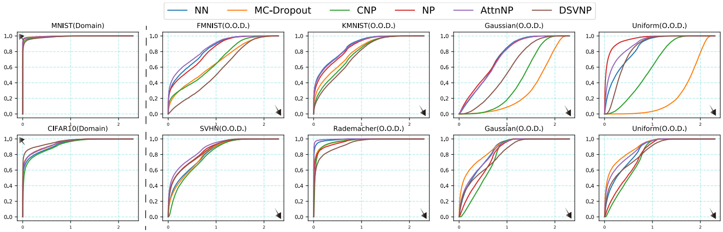
Metrics	CNP	NP	AttnNP	DSVNP
NLL	-2.014 (9E-4)	-1.537 (1E-3)	-1.821 (7E-3)	**-2.145 (9E-4)**
MSE	0.096 (3E-4)	0.074 (2E-4)	0.067 (1E-4)	**0.036 (2.1E-5)**

Table 4. Predictive MSEs on Multi-Output Dataset. CNP's results are for target points. D records (input,output) dimensions, and N is the number of samples. MC-Dropout runs 50 stochastic forward propagation and average results for prediction in each data point. (Figures in brackets are variances.)

Dataset	N	D	MC-Dropout	CNP	NP	AttnNP	DSVNP
Sarcos	48933	(21,7)	1.215(3E-3)	1.437(2.9E-2)	1.285(1.2E-1)	1.362(8.4E-2)	**0.839(1.5E-2)**
WQ	1060	(16,14)	0.007(9.6E-8)	0.015(2.4E-5)	0.007(5.2E-6)	0.01(8.5E-6)	**0.006(1.6E-6)**
SCM20D	8966	(61,16)	0.017(2.4E-7)	0.037(6.7E-5)	0.015(7.1E-8)	0.015(8.1E-7)	**0.007(2.3E-7)**

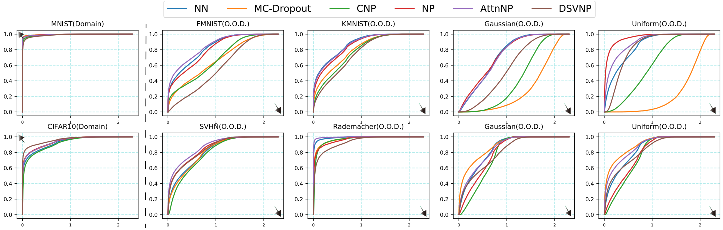
Classification with Uncertainty Quantification

Observations in image classification and out of distribution detection (based on cumulative distribution of entropies) :



Classification with Uncertainty Quantification

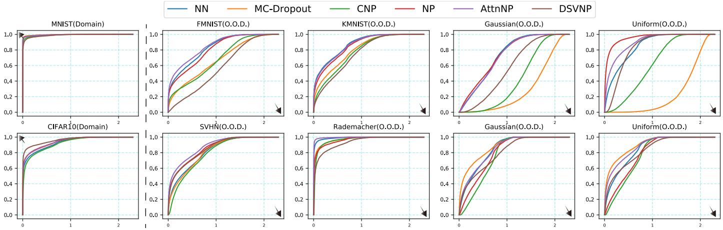
Observations in image classification and out of distribution detection (based on cumulative distribution of entropies) :



- MNIST: no significant difference in classification performance/o.o.d detection (all above 99%) ; DSVNP $\rightarrow$ better o.o.d. detection on FMNIST/KMNIST ; MC-D more robust to Gaussian/Uniform noise.

Classification with Uncertainty Quantification

Observations in image classification and out of distribution detection (based on cumulative distribution of entropies) :



- MNIST: no significant difference in classification performance/o.o.d detection (all above 99%) ; DSVNP → better o.o.d. detection on FMNIST/KMNIST ; MC-D more robust to Gaussian/Uniform noise.
- CIFAR10: DSVNP(86.3%) ≻ MC/CNP ≻ AttnNP/NP ≻ NN (Classification Performance) ; DSVNP → best entropy distributions in domain dataset and most robust to Rademacher noise.

Future Works

Some Challenging and Promising Directions

- More effective inference methods for our proposed hierarchical $\mathcal{SP}$s

Some Challenging and Promising Directions

- More effective inference methods for our proposed hierarchical $\mathcal{SP}$s
- More expressive context latent variable using higher order statistics

Some Challenging and Promising Directions

- More effective inference methods for our proposed hierarchical $\mathcal{SP}$s
- More expressive context latent variable using higher order statistics
- More explorations to Uncertainty-aware Decision-making Problems

Thanks for Your Listening

M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. J. Rezende, S. Eslami, and Y. W. Teh, "Neural processes," *arXiv preprint arXiv:1807.01622*, 2018.

M. Deisenroth and C. E. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *Proceedings of the 28th International Conference on machine learning (ICML-11)*, 2011, pp. 465–472.

F. P. Casale, A. Dalca, L. Saglietti, J. Listgarten, and N. Fusi, "Gaussian process prior variational autoencoders," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 369–10 380.

D. Heath and W. Sudderth, "De finetti's theorem on exchangeable variables," *The American Statistician*, vol. 30, no. 4, pp. 188–189, 1976.

S. Park and S. Choi, "Hierarchical gaussian process regression," in *Proceedings of 2nd Asian Conference on Machine Learning*, 2010, pp. 95–110.

A. Damianou and N. Lawrence, "Deep gaussian processes," in *Artificial Intelligence and Statistics*, 2013, pp. 207–215.

Z. Dai, A. Damianou, J. González, and N. Lawrence, "Variational auto-encoded deep gaussian processes," *arXiv preprint arXiv:1511.06455*, 2015.

D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

Y. Gal, R. McAllister, and C. E. Rasmussen, "Improving pilco with bayesian neural network dynamics models," in *Data-Efficient Machine Learning workshop, ICML*, vol. 4, 2016, p. 34.