Gaussian Processes

Presenter: Zhe Wang

https://qdata.github.io/deep2Read

Zhe Wang

201909

Content

Definition of stochastic processes:

Suppose that (Ω, F, P) is a probability space, and that $I \subset R$ is of infinite cardinality. Suppose further that for each $x \in I$, there is a random variable $f_x : \Omega \to R$. The function $f : I \times \Omega \to R$ defined by $f(x, w) = f_x(w)$ is called a stochastic process with indexing set I, and is written $f = \{f_x, x \in I\}$.

Definition of Gaussian processes:

Gaussian process is a stochastic process, such that every finite collection of those random variables has a multivariate normal distribution. Suppose $I = \{x_1, \cdots, x_n\}$, and f is a Gaussian processes defined on I, then suppose $\hat{I} \subset I$, we have $f_{\hat{I}}$ is a multivariate normal distribution.

Each GP is specified by mean and covariance function, if $f(x)$ is a GP with:

$$E(f(x)) = \mu(x), \quad K(x, x') = cov(f(x), f(x')),$$

then $f(x)$ can be denoted as $f(x) \sim GP(\mu(x), K(x, x'))$.

It is also worth noticing that a Gaussian distribution also has the Gaussian marginal distribution and conditional distribution.

Assume that $f(x) = \Phi(x)^T w$, with prior $w \sim N(0, \Sigma_p)$. $\Phi(x)$ projects x into a higher dimensional reproducing kernel hilbert space(RKHS). For mean and covariance,

$$\begin{aligned} \mu(x) &= E[f(x)] = \Phi(x)^T E(w) = 0, \\ cov(x, x') &= E[f^T(x) f(x')] = \Phi(x)^T \Sigma_p \Phi(x') \end{aligned} \tag{1}$$

Since Σ_p is a SPD matrix, $\Phi(x)^T \Sigma_p \Phi(x')$ is also a SPD matrix, and can be expressed by $K(x, x')$, where K is a kernel funciton

Given n pairs of observations $\{x_i, y_i\}_{i=1}^n$, we are trying to predict $y^* = f(X^*)$ at the location $X^* = (x_1^*, , x_m^*)$. Assume the stochastic process f follows $GP(0, K)$:

$$[y, y^*] = [f(x_1), \cdots, f(x_n), f(x_1^*), \cdots, f(x_m^*)] = N(0, K),$$

where K is the noisy kernel matrix whose ij-element is given by $K_{ij} = K(x_i, x_j)$.

The joint distribution over observed target y and the predicted target y^* is given by:

$$\begin{bmatrix} y \\ y^* \end{bmatrix} \sim N(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K(X,X), K(X,X^*) \\ K(X^*,X), K(X^*,X^*) \end{bmatrix}),$$

where $K(X^*, X)$ is a $m \times n$ matrix of which the i,j-element $K(X^*, X)_{i,j} = K(x_i^*, x_j)$ and $K(X^*, X) = K(X, X^*)^T$.

Taking advantages of characteristics of Gaussian distribution,

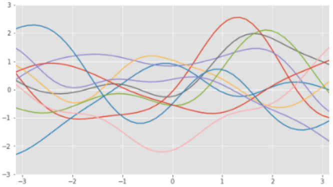
$$
\begin{aligned}
&p(y^*|X^*, X, y) = N(\hat{\mu}, \hat{\Sigma}) \\
&\hat{\mu} = K(X^*, X)K(X, X)^{-1}y \\
&\hat{\Sigma} = K(X^*, X^*) - K(X^*, X)K(X, X)^{-1}K(X, X^*)
\end{aligned}
$$

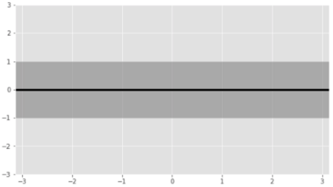
Bayesian Analysis:

The prior over the joint distribution is $P(y, y^*) \sim N(0, K)$, inheriting the property of Gaussian distribution, the posterior $P(y^*|x^*, X, y) \sim N(\hat{\mu}, \hat{\Sigma})$

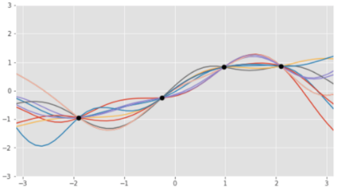
There are many functions $f : X \rightarrow Y, s.t.\ f(X) = Y$, and GP is a distribution over all possible functions and is able to output the probability of each function.

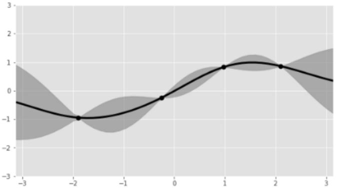
Experiment results:
Sampling from prior distribution





Sampling from posterior distribution





Conditional Neural Processes

Marta Garnelo, Dan Rosenbaum, Chris J. Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo J. Rezende, S. M. Ali Eslami

DeepMind, Imperial College London

Motivation:

- Deep neural network are trained from scratch for each new function
- Bayesian methods, such as GPs, exploit prior knowledge to quickly infer the shape of a new function at test time
- Computationally expensive during test time and hard to design priors and proper kernel function.

Contribution:

- GP inspired neural network, enable few shot learning
- Output the uncertainty for each prediction.
- Scale to large datasets and complex functions.

Main idea

CNP is modelling the posterior $P(Y_T|X_T, X_O, Y_O)$. Intuitively, the model learns experience from (X_O, Y_O) and embeds the knowledge in feature prediction.

Main idea
The experience in introduced by another variable Z.

$$Z = h(X_O, Y_O), \quad P(f(X_T)|X_T, Z) \tag{2}$$

Suppose $O = \{(x_i, y_i)\}_{i=0}^{n-1} \subset X \times Y$ of pairs of inputs $x_i \in X$ and outputs $y_i \in Y$ and another set $T = \{x_i\}_{i=n}^{n+m-1} \subset X$ of unlabelled points.

Let P be a probability distribution over functions $f : X \to Y$, formally known as a stochastic process , then for $f \sim P$, set $y_i = f(x_i)$.

P defines a joint distribution over the random variables $\{f(x_i)\}_{i=0}^{n+m-1}$, and therefore a conditional distribution $P(f(T)|O, T)$; our task is to predict the output values $f(x)$ for every $x \in T$ given O.

Model

CNP is a a conditional stochastic process Q_θ that defines distributions over $f(x)$ for inputs $x \in T$

$$P(f(T)|T, O) \approx Q_\theta(f(T)|T, O)$$

Some properties of Stochastic Processes:

- Permutation invariance: Q_θ is invariant to permutations of O and T. of O and T.

$$Q_\theta(f(T)|O, T) = Q_\theta(f(T')|O, T') = Q_\theta(f(T)|O', T).$$

In order for the permutation invariance w.r.t. T, we consider Q_θ that factor

$$Q_\theta(f(T)|O, T) = \prod_{x \in T} Q_\theta(f(x)|O, x)$$

Model

The defining characteristic of a CNP is that it conditions on O via an embedding of fixed dimensionality. The designing of the architecture:

$$\begin{aligned} r_i &= h_\theta(x_i, y_i), \quad \forall (x_i, y_i) \in O \\ r &= r_1 \oplus r_2 \oplus ... r_{n-1} \oplus r_n \\ \phi_i &= g_\theta(x_i, r), \quad \forall (x_i) \in T \end{aligned}$$

- where $h_\theta : X \times Y \rightarrow R^d$ and $g_\theta : X \times R^d \rightarrow R^e$ are neural networks.
- In order for the permutation invariance in O, the operator $\oplus$ is commutative.
- Φ_i are parameters for $Q_\theta(f(x_i)|O, x_i) = Q(f(xi)|\Phi_i)$.

- For regression tasks we use Φ_i to parametrize the mean and variance $\Phi_i = (\mu_i, \sigma_i^2)$ of a Gaussian distribution $N(\mu_i, \sigma_i^2)$ for every $x_i \in T$
- For classification tasks Φ_i parametrizes the logits of the class probabilities p_c over the c classes of a categorical distribution
- we take $a_1 \oplus \cdots \oplus a_n$ to be the mean operation $(a_1 + \cdots + a_n)/n$

Training:
Suppose $O = \{(x_1, y_1), \cdots, (x_n, y_n)\}$. During training, r is generated from O_N, a subset of O, the loss is defined as:

$$L(\theta) = -E_{f\sim P}[E_N[\log Q_\theta(\{y_i\}_{i=0}^{n-1} | O_N, \{x_i\}_{i=0}^{n-1})]]$$

Function regression:

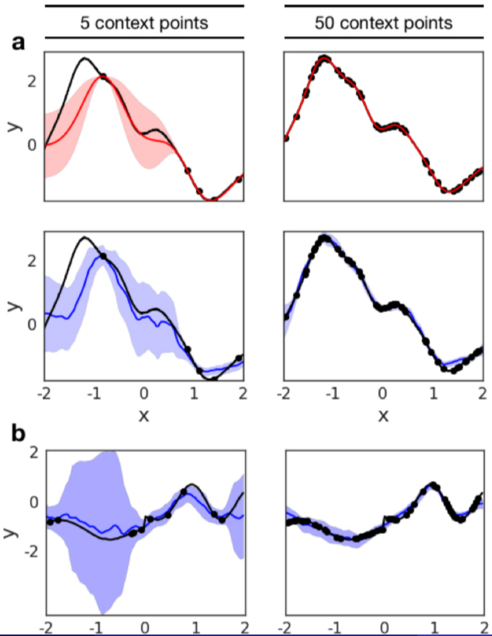
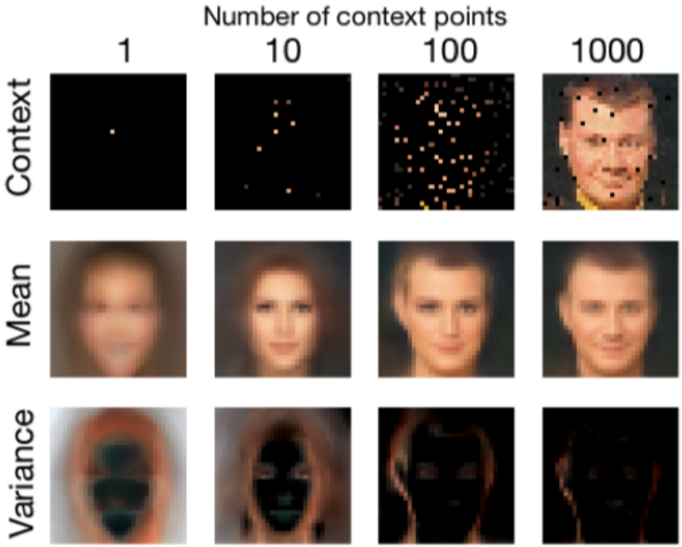


Image completion:
image completion as a regression task over functions in either $f : [0,1]^2 \rightarrow [0,1]$ for grayscale images, or $f : [0,1]^2 \rightarrow [0,1]^3$ for RGB images. The input x is the 2D pixel coordinates normalized to $[0,1]^2$, and the output y is either the grayscale intensity or a vector of the RGB intensities of the corresponding pixel.

CelebA


Number of context points
1
10
100
1000
Context
Mean
Variance

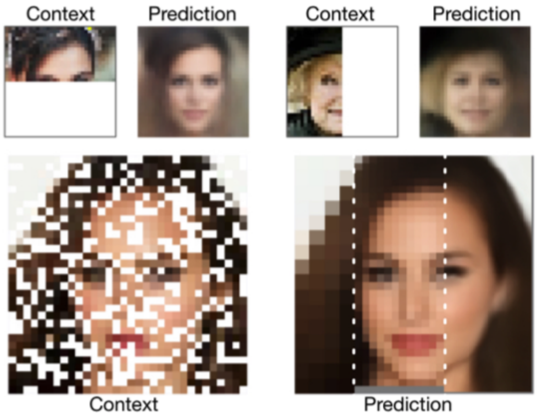


Figure 5. **Flexible image completion**. In contrast to standard conditional models, CNPs can be directly conditioned on observed pixels in arbitrary patterns, even ones which were never seen in the training set. Similarly, the model can predict values for pixel coordinates that were never included in the training set, like subpixel values in different resolutions. The dotted white lines were added for clarity after generation.

Few shot classification:
the setting and dataset is similar to meta learning.

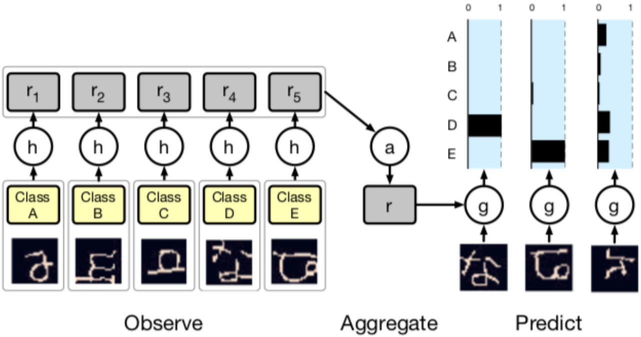
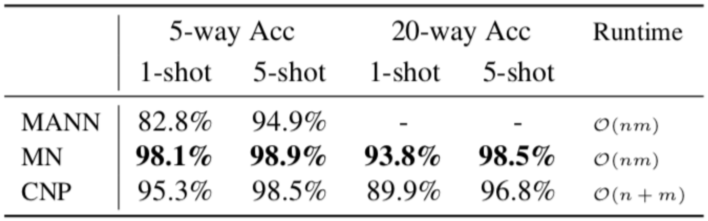


Figure 7. **One-shot Omniglot classification.** At test time the model is presented with a labelled example for each class, and outputs the classification probabilities of a new unlabelled example. The model uncertainty grows when the new example comes from an un-observed class.



	5-way Acc		20-way Acc		Runtime
	1-shot	5-shot	1-shot	5-shot	
MANN	82.8%	94.9%	-	-	$\mathcal{O}(nm)$
MN	**98.1%**	**98.9%**	**93.8%**	**98.5%**	$\mathcal{O}(nm)$
CNP	95.3%	98.5%	89.9%	96.8%	$\mathcal{O}(n + m)$

Table 2. **Classification results on Omniglot**. Results on the same task for MANN (Santoro et al., 2016), and matching networks (MN) (Vinyals et al., 2016) and CNP.

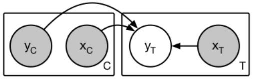
Neural Processes

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum Fabio Viola, Danilo J. Rezende, S. M. Ali Eslami, Yee Whye Teh

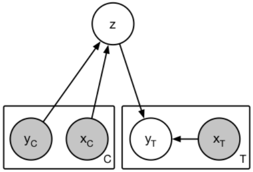
DeepMind

Motivation: a variational version of CNP.
For the same context data, CNPs are unable to produce different function samples, which can be important if modelling this uncertainty is desirable.
The data need different explanations to help decision making, especially under limited data scenarios.

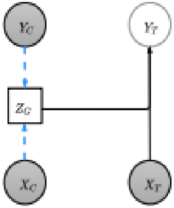
The NP is a generalisation of CNP. They are both model-inspired deep learning architectures.



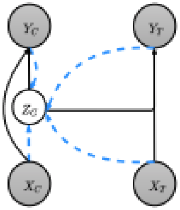
y_C
x_C
C
y_T
x_T
T



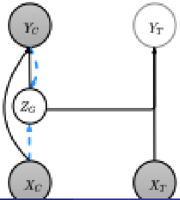
z
y_C
x_C
C
y_T
x_T
T



Y_C
Y_T
Z_C
X_C
X_T

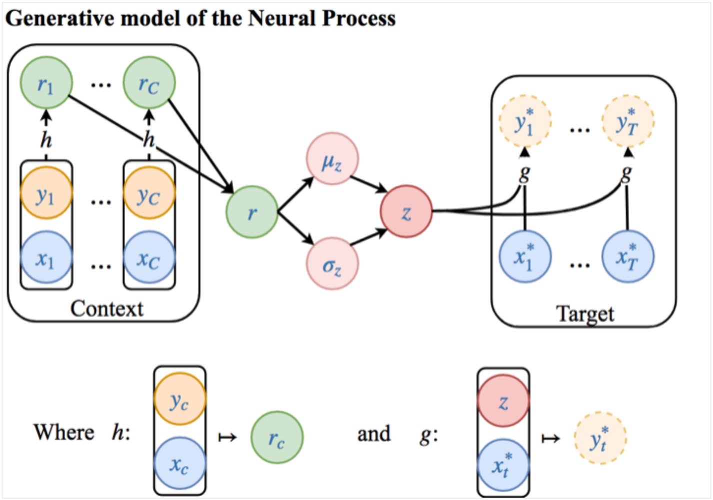


Y_C
Y_T
Z_C
X_C
X_T



Y_C
Y_T
Z_C
X_C
X_T

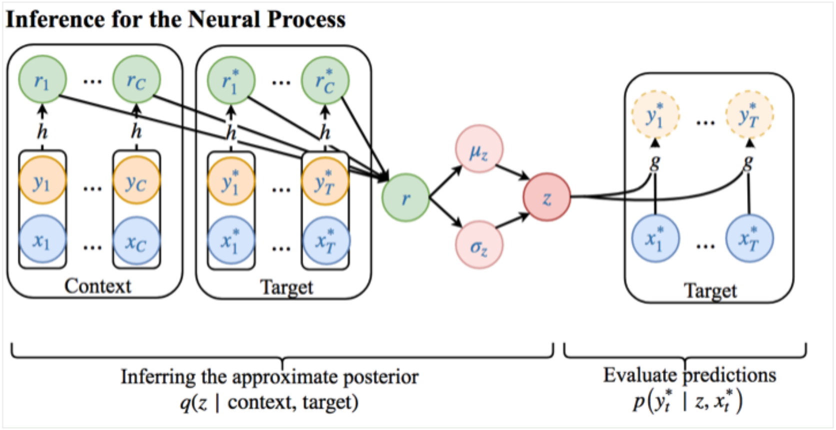
Generative process:


Generative model of the Neural Process
r_1
r_C
h
h
y_1
y_C
x_1
x_C
Context
r
μ_z
σ_z
z
y_1^*
y_T^*
g
g
x_1^*
x_T^*
Target
Where h: y_c, x_c $\mapsto$ r_c
and g: z, x_t^* $\mapsto$ y_t^*

Generative process

- $(X_C, Y_C) \rightarrow r$
- $r \rightarrow (\mu_z, \sigma_z)$
- $(\mu_z, \sigma_z) \rightarrow z$
- $[z, x_i^*] \rightarrow y_i^*$

Inference process:

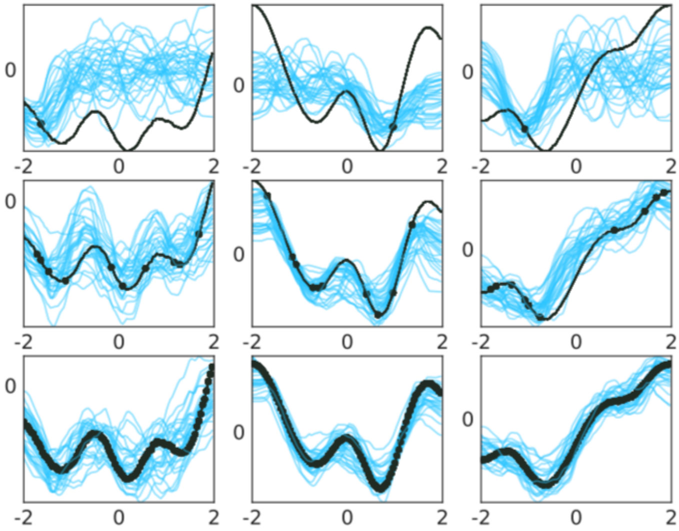


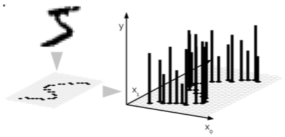
Inference process:

- $([X_C, X_T], [Y_C, Y_T]) \rightarrow r$
- $r \rightarrow (\mu_z, \sigma_z)$
- $(\mu_z, \sigma_z) \rightarrow z$

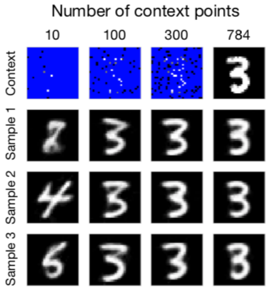
ELBO:

$$E_{q(z|[X_C,X_T],[Y_C,Y_T])}[\sum_{t=1}^{T}[log(y_t * |x_t^*, z)]] - KL(q_z||p_z)$$

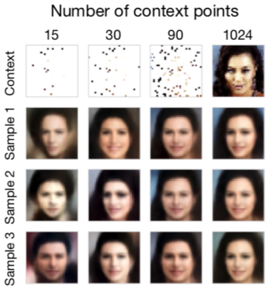

0
-2
0
2



y
x1
x0



Number of context points
10
100
300
784
Context
Sample 1
Sample 2
Sample 3



Number of context points
15
30
90
1024
Context
Sample 1
Sample 2
Sample 3