Gaussian Process Regression

Qi Wang

January 4, 2021

Overview

1. Gaussian Process
2. Gaussian Process Regression
3. Algorithm
   - Cholesky factorization
   - Sparse Gaussian Processes Regression (SGPR)
4. Neural Network v.s. Gaussian Process
5. Numerical Test

Gaussian Process

Definition 1

A Gaussian process is a collection of random variables, i.e., $\{f(x_1), \cdots, f(x_n)\}$, any finite number of which have a joint Gaussian distribution and $f(x)$ is a function of x where $x \in \mathbb{R}^d$. We can write the Gaussian process as

$$\begin{pmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{pmatrix} \sim \mathcal{GP}\left(\begin{pmatrix} \mu(x_1) \\ \mu(x_2) \\ \vdots \\ \mu(x_n) \end{pmatrix}, \begin{pmatrix} k(x_1, x_1) & \cdots & k(x_1, x_n) \\ k(x_2, x_1) & \cdots & k(x_2, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_n) \end{pmatrix} \right)$$

where $\mu(x_i)$ is prior mean function of $f(x_i)$ (usually constant, e.g., 0), and $k(x_i, x_j)$ is prior covariance function of (x_i, x_j) (approximating covariance of $(f(x_i), f(x_j))$.

Gaussian Process

notation	meaning	size
X_1	training inputs	$\mathbb{R}^{n_1 \times d}$
$f(X_1)$	training targets	$\mathbb{R}^{n_1}$
X_2	test inputs	$\mathbb{R}^{n_2 \times d}$
$f(X_2)$	test targets	$\mathbb{R}^{n_2}$
μ_A	mean vector, where $(\mu_A)_i = \mu(A_i)$	the same with $f(A)$
Σ_{AB}	covariance matrix, where $(\Sigma_{AB})_{ij} = k(A_i, B_j)$	size of $f(A)\times$ size of $f(B)$

Table: Notation

Then,

$$\begin{pmatrix} f(X_1) \\ f(X_2) \end{pmatrix} \sim \mathcal{GP}(\begin{pmatrix} \mu_{X_1} \\ \mu_{X_2} \end{pmatrix}, \begin{pmatrix} \Sigma_{X_1X_1} & \Sigma_{X_1X_2} \\ \Sigma_{X_2X_1} & \Sigma_{X_2X_2} \end{pmatrix})$$

Gaussian Process Regression

The target of machine learning problem is to calculate the realization of the posterior random variable $f(X_2)|f(X_1)$. By marginalization property and Bayes conditional probability, we have

$$f(X_2)|f(X_1) \sim \mathcal{N}(\mu_{X_2} + \Sigma_{X_2X_1}\Sigma_{X_1X_1}^{-1}(f(X_1) - \mu_{X_1}), \\ \Sigma_{X_2X_2} - \Sigma_{X_2X_1}\Sigma_{X_1X_1}^{-1}\Sigma_{X_1X_2}).$$

For a single test input, $x_i \in X_2$, the mean $\bar{\mu}$ of $f(x_i)|f(X_1)$ is the test target prediction and with its variance $\bar{k}$ we can calculate the confident interval. For example, $(\bar{\mu}_i - 2\sqrt{\bar{k}_i}, \bar{\mu}_i + 2\sqrt{\bar{k}_i})$ corresponds 95% confident interval.

Train set with noise

For $(x_i, y_i) \in (X_1, Y_1)$, with noise $\epsilon_i \in \mathcal{N}(0, \sigma^2)$ being independent with $f(x_i)$, we have

$$y_i = f(x_i) + \epsilon_i$$

For $i \in \mathbb{Z}\, \epsilon_i$ are iids. The covariance of (y_i, y_j) becomes

$$\begin{aligned} cov(y_i, y_j) =& \mathbb{E}[(f(x_i) + \epsilon_i - \mu(x_i))(f(x_j) + \epsilon_j - \mu(x_j))] \\ =& \mathbb{E}[(f(x_i) - \mu(x_i))(f(x_j) - \mu(x_j)) + \epsilon_i(f(x_j) - \mu(x_j)) \\ & + \epsilon_j(f(x_i) - \mu(x_i)) + \epsilon_i\epsilon_j] \\ =& \begin{cases} k(x_i, x_j), & \text{if } i \neq j, \\ k(x_i, x_j) + \sigma^2, & \text{if } i = j. \end{cases} \end{aligned}$$

Train set with noise

Then the distribution of Gaussian process $(Y_1, f(X_2))$ is

$$\begin{pmatrix} Y_1 \\ f(X_2) \end{pmatrix} \sim \mathcal{GP}(\begin{pmatrix} \mu_{X_1} \\ \mu_{X_2} \end{pmatrix}, \begin{pmatrix} \Sigma_{X_1 X_1} + \sigma^2 I_{n_1} & \Sigma_{X_1 X_2} \\ \Sigma_{X_2 X_1} & \Sigma_{X_2 X_2} \end{pmatrix}).$$

And the mean and covariance of posterior r.v. $f(X_2)|Y_1$ is

$$\tilde{\mu} = \mu_{X_2} + \Sigma_{X_2 X_1}(\Sigma_{X_1 X_1} + \sigma^2 I_{n_1})^{-1}(Y_1 - \mu_{X_1}),$$
$$\tilde{\Sigma} = \Sigma_{X_2 X_2} - \Sigma_{X_2 X_1}(\Sigma_{X_1 X_1} + \sigma^2 I_{n_1})^{-1}\Sigma_{X_1 X_2}$$

covariance function - kernel

A covariance function is a kernel $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. Variant kernels:

- RBF kernel (stationary)

$$k(x_1, x_2) = \sigma^2 \exp(-\frac{\|x_1 - x_2\|^2}{2l^2})$$

- Periodic kernel (stationary)

$$k(x_1, x_2) = \sigma^2 \exp(-\frac{2\sin^2(\pi\|x_1 - x_2\|/p)}{l^2})$$

- Linear kernel (non-stationary)

$$k(x_1, x_2) = \sigma_b^2 + \sigma^2 (x_1 - c)^T (x_2 - c)$$

- compositing kernels by addition and multiplication on old kernels.

Hyperparameters can be tuned by optimizing marginal likelihood or log marginal likelihood of train data set (if with noisy)

$$\log p(Y_1|X_1) = -\frac{1}{2} Y_1^T (\Sigma_{X_1X_1} + \sigma^2 I_{n_1})^{-1} Y_1 - \frac{1}{2} \log|\Sigma_{X_1X_1} + \sigma^2 I_{n_1}| - \frac{n_1}{2} \log 2\pi$$

which derived by the distribution of $Y_1 \sim \mathcal{GP}(0, \Sigma_{X_1X_1} + \sigma^2 I_{n_1})$.

Summary

Cons

- The $n \times n$ matrix inversion operation is $\mathcal{O}(n^3)$ time complexity.
- The covariance matrix needs storage as $\mathcal{O}(n^2)$ where n is the number of training examples.

Pros

- Give distribution for the prediction value and confidence interval.
- Easily interpolate training data.

Algorithm-Cholesky factorization

Algorithm 1: Gaussian processes regression (Cholesky factorization)

Input: training dataset $(X_1, Y_1) \in (\mathbb{R}^{n_1 \times d}, \mathbb{R}^{n_1})$; kernel function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$; train dataset noise standard deviation σ; single test data $\mathbf{x} \in \mathbb{R}^d$

Output: target prediction $\tilde{\mu}$; target variance $\tilde{\Sigma}$

1. Calculate covariance matrix of train dataset $\Sigma_{X_1X_1}$ where $(\Sigma_{X_1X_1})_{ij} = k(\mathbf{x_i}, \mathbf{x_j}), i, j \in \{1, \cdots, n_1\}$
2. Calculate covariance vector Σ_{xX_1} of x and X_1 where $(\Sigma_{xX_1})_i = k(\mathbf{x}, \mathbf{x_i}), i \in \{1, \cdots, n_1\}$
3. Calculate variance of test data $\Sigma_{xx} = k(\mathbf{x}, \mathbf{x})$
4. $L :=$ cholesky$(\Sigma_{X_1X_1} + \sigma^2 I)$
5. Solve $L\mathbf{a} = Y_1$ for $\mathbf{a}$ by forward substitution, and solve $L^T\mathbf{b} = \mathbf{a}$ for $\mathbf{b}$ by back substitution.
6. $\tilde{\mu} = \Sigma_{xX_1}^T \mathbf{b}$
7. Solve $L\mathbf{c} = \Sigma_{xX_1}$ for $\mathbf{c}$ by forward substitution.
8. $\tilde{\Sigma} = \Sigma_{xx} - \mathbf{c}^T\mathbf{c}$

Sparse Gaussian Processes Regression (SGPR)

Key idea: A smaller sampled training set (X_m, Y_m) from (X_1, Y_1), called inducing sets or "active set" or "pseudo-inputs", where $m \ll n$.

Assumption 1

Conditional r.v.s $f(X_2)|f(X_m)$ and $f(X_1)|f(X_m)$ are independent given $f(X_m)$.

Assume $f(X_m) = Y_m \sim \mathcal{N}(\mathbf{0}, \Sigma_{X_m X_m})$ (noise free)

$$
\begin{aligned}
p(f(X_1), f(X_2)) &= \int p(f(X_1), f(X_2), f(X_m)) d(f(X_m)) \\
&= \int p(f(X_2), f(X_1)|f(X_m)) p(f(X_m)) d(f(X_m)) \\
&= \int p(f(X_2)|f(X_m)) p(f(X_1)|f(X_m)) p(f(X_m)) d(f(X_m)) \\
&\approx \int q(f(X_2)|f(X_m)) q(f(X_1)|f(X_m)) p(f(X_m)) d(f(X_m)),
\end{aligned}
$$

where $q(.)$ is the approximation pdf of $p(.)$.

Sparse Gaussian Processes Regression (SGPR)

$$
\begin{aligned}
p(f(X_1), f(X_2)|Y_1) &= \frac{p(f(X_1), f(X_2))p(Y_1|f(X_1))}{p(Y_1)} \\
p(f(X_2)|Y_1) &= \int p(f(X_1), f(X_2)|Y_1)d(f(X_1)) \\
&= \frac{1}{p(Y_1)} \int p(Y_1|f(X_1))p(f(X_1), f(X_2))d(f(X_1))
\end{aligned}
$$

where $p(Y_1|f(X_1)) = \mathcal{N}(f(X_1), \sigma^2 I)$

SGPR - The Deterministic Training Conditional (DTC) Approximation

$$
\begin{aligned}
q_{DTC}(f(X_1)|f(X_m)) &= \mathcal{N}(\Sigma_{X_1X_m}\Sigma_{X_mX_m}^{-1}f(X_m), \mathbf{0}) \quad \text{(deterministic)} \\
q_{DTC}(f(X_2)|f(X_m)) &= p(f(X_2)|f(X_m)) \\
&= \mathcal{N}(\Sigma_{X_2X_m}\Sigma_{X_mX_m}^{-1}f(X_m), \Sigma_{X_2X_2} - \mathcal{Q}_{X_2X_2})
\end{aligned}
$$

where $\mathcal{Q}_{AB} := \Sigma_{AX_m}\Sigma_{X_mX_m}^{-1}\Sigma_{X_mB}$. Thus

$$
\begin{pmatrix} f(X_1) \\ f(X_2) \end{pmatrix}_{DTC} \sim \mathcal{N}(\begin{pmatrix} \mathbf{0} \\ \mathbf{0} \end{pmatrix}, \begin{pmatrix} \mathcal{Q}_{X_1X_1} & \mathcal{Q}_{X_1X_2} \\ \mathcal{Q}_{X_2X_1} & \Sigma_{X_2X_2} \end{pmatrix})
$$

And the mean and covariance of posterior r.v. $f(X_2)|Y_1$ (with noise) are

$$
\begin{aligned}
\tilde{\mu}_{DTC} &= \mathcal{Q}_{X_2X_1}(\mathcal{Q}_{X_1X_1} + \sigma^2 I_{n_1})^{-1}Y_1 = \sigma^{-2}\Sigma_{X_2X_m}\Theta\Sigma_{X_mX_1}Y_1, \\
\tilde{\Sigma}_{DTC} &= \Sigma_{X_2X_2} - \mathcal{Q}_{X_2X_1}(\mathcal{Q}_{X_1X_1} + \sigma^2 I_{n_1})^{-1}\mathcal{Q}_{X_1X_2} \\
&= \Sigma_{X_2X_2} - \mathcal{Q}_{X_2X_2} + \Sigma_{X_2X_m}\Theta\Sigma_{X_mX_2}.
\end{aligned}
$$

where $\Theta := (\sigma^{-2}\Sigma_{X_mX_1}\Sigma_{X_1X_m} + \Sigma_{X_mX_m})^{-1}$ has rank (at most) m.

SGPR - The Deterministic Training Conditional (DTC) Approximation

Prove $\mathcal{Q}_{X_2X_1}(\mathcal{Q}_{X_1X_1} + \sigma^2 I_{n_1})^{-1} Y_1 = \sigma^{-2}\Sigma_{X_2X_m}\Theta\Sigma_{X_mX_1} Y_1$

Proof.

$$
\begin{aligned}
&\sigma^{-2}\Sigma_{X_2X_m}\Theta\Sigma_{X_mX_1} Y_1 \\
&= \sigma^{-2}\Sigma_{X_2X_m}(\sigma^{-2}\Sigma_{X_mX_1}\Sigma_{X_1X_m} + \Sigma_{X_mX_m})^{-1}\Sigma_{X_mX_1} Y_1 \\
&= \Sigma_{X_2X_m}(\Sigma_{X_mX_1}\Sigma_{X_1X_m} + \sigma^2\Sigma_{X_mX_m})^{-1}\Sigma_{X_mX_1} Y_1 \\
&= \Sigma_{X_2X_m}\Sigma_{X_mX_m}^{-1}(\Sigma_{X_mX_1}\Sigma_{X_1X_m}\Sigma_{X_mX_m}^{-1} + \sigma^2 I)^{-1}\Sigma_{X_mX_1} Y_1 \\
&= \Sigma_{X_2X_m}\Sigma_{X_mX_m}^{-1}\Sigma_{X_mX_1}(\Sigma_{X_mX_1}\Sigma_{X_1X_m}\Sigma_{X_mX_m}^{-1}\Sigma_{X_mX_1} + \sigma^2\Sigma_{X_mX_1})^{-1}\Sigma_{X_mX_1} Y_1 \\
&= \Sigma_{X_2X_m}\Sigma_{X_mX_m}^{-1}\Sigma_{X_mX_1}(\Sigma_{X_1X_m}\Sigma_{X_mX_m}^{-1}\Sigma_{X_mX_1} + \sigma^2 I)^{-1} Y_1 \\
&= \mathcal{Q}_{X_2X_1}(\mathcal{Q}_{X_1X_1} + \sigma^2 I_{n_1})^{-1} Y_1
\end{aligned}
$$

$\square$

SGPR - The Fully Independent Training Conditional (FITC) Approximation

$$
\begin{aligned}
q_{FITC}(f(X_1)|f(X_m)) &= \prod_{i=1}^{n_1} p(f(x_i)|f(X_m)) \quad \text{(independent conditionals)} \\
&= \mathcal{N}(\Sigma_{X_1X_m}\Sigma_{X_mX_m}^{-1} f(X_m), \mathrm{diag}[\Sigma_{X_1X_1} - \mathcal{Q}_{X_1X_1}]) \\
q_{FITC}(f(X_2)|f(X_m)) &= p(f(X_2)|f(X_m))
\end{aligned}
$$

Thus

$$
\begin{pmatrix} f(X_1) \\ f(X_2) \end{pmatrix}_{FITC} \sim \mathcal{N}(\begin{pmatrix} \mathbf{0} \\ \mathbf{0} \end{pmatrix}, \begin{pmatrix} \mathcal{Q}_{X_1X_1} - \mathrm{diag}(\mathcal{Q}_{X_1X_1} - \Sigma_{X_1X_1}) & \mathcal{Q}_{X_1X_2} \\ \mathcal{Q}_{X_2X_1} & \Sigma_{X_2X_2} \end{pmatrix}).
$$

SGPR - The Fully Independent Training Conditional (FITC) Approximation

And the mean and covariance of posterior r.v. $f(X_2)|Y_1$ (with noise) are

$$\tilde{\mu}_{FITC} = \mathcal{Q}_{X_2X_1}(\mathcal{Q}_{X_1X_1} + \Lambda)^{-1}Y_1, = \Sigma_{X_2X_m}\Theta_2\Sigma_{X_mX_1}\Lambda^{-1}Y_1$$

$$\begin{aligned}\tilde{\Sigma}_{FITC} &= \Sigma_{X_2X_2} - \mathcal{Q}_{X_2X_1}(\mathcal{Q}_{X_1X_1} + \Lambda)^{-1}\mathcal{Q}_{X_1X_2} \\ &= \Sigma_{X_2X_2} - \mathcal{Q}_{X_2X_2} + \Sigma_{X_2X_m}\Theta_2\Sigma_{X_mX_2},\end{aligned}$$

where $\Lambda := \mathrm{diag}(\Sigma_{X_1X_1} - \mathcal{Q}_{X_1X_1} + \sigma^2 I_{n_1})$ and
$\Theta_2 := (\Sigma_{X_mX_m} + \Sigma_{X_mX_1}\Lambda^{-1}\Sigma_{X_1X_m})^{-1}$ has rank (at most) m.

Neural Network v.s. Gaussian Process

Theorem 2

Neural networks with infinite width (infinite number of hidden units) is equivalent to GPs.

In a M-layer neural network, for each layer l,

$$x_j^l(x) = \phi(z_j^{l-1}(x)), \quad z_i^l(x) = b_i^l + \sum_{j=1}^{N_l} W_{ij}^l x_j^l(x),$$

where ϕ represents activation functions. Since prior parameters b_i^l and W_{ij}^l are all i.i.ds, by Central Limit Theorem, $z_i^l(x)$ is Gaussian distributed when $N_l \to \infty$.

Therefore, $z_i^l \sim \mathcal{GP}(\mu^l, k^l)$ with $\mu^l = 0$ and

$$k^l(x_p, x_q) = E(z_i^l(x_p) z_i^l(x_q)) = \sigma_b^2 + \sigma_w^2 E(x_i^l(x_p) x_i^l(x_q)).$$

When $M = 1$, Neal(1996) shows that $E(x_i^1(x_p) x_i^1(x_q))$ can be obtained by integrating over W^0 and b^0.

Numerical Test Set up

- Dataset: Combined Cycle Power Plant Data Set from UCI [1]: 9568 samples, 4 attributes, and 1 target. Attributes are Temperature, Ambient Pressure, Relative Humidity and Exhaust Vacuum. The target is Net hourly electrical energy output.
- Randomly split the dataset by 7 : 3. Calculated train target Y_1's mean and std μ_{Y_1}, σ_{Y_1}, and normalized Y_1 by $(Y_1 - \mu_{Y_1})/\sigma_{Y_1}$ so that it has normal distribution $\mathcal{N}(0, 1)$ and use μ_{Y_1}, σ_{Y_1} to recover the predict test target, i.e. $f(X_2)_{\textbf{predict}} * \sigma_{Y_1} + \mu_{Y_1}$.
- Run Algorithm 1 using python Sklearn package.
- Run GPR, SGPR using python GPy package.
- Run DNN using tensorflow (with Adam optimizer, MSE loss, and learning rate 0.005).
- For each setup, repeat experiment 5 times. Evaluated the root mean square error (RMSE) of test data. The RMSE is

$$RMSE = \sqrt{\sum_{i=1}^{n_2} (Y_2^i - f(X_2^i))^2}.$$

Numerical Test - GPR (SKlearn) Result

Table: GPR test results: the column 'time(s)' is the total time (train plus test) of 5 repetitions.

noise (σ^2)	kernel	is_optimization	RMSE		time(s)
			mean	std	
0.001	linear	no	4.6148	0.0403	2629.5757
0.1	linear	no	4.6145	0.0406	3492.5053
0.001	linear	yes	4.6148	0.0403	4698.1432
0.1	linear	yes	4.6145	0.0406	3506.6333
0.001	RBF	no	16.915	0.1299	235.6588
0.1	RBF	no	16.3234	0.0406	645.5147
0.001	RBF	yes	14.776	4.3117	1207.4654
0.1	RBF	yes	4.0863	0.0296	1619.0353
0.001	Periodic	no	failed (nonpositive definite matrix)		
0.1	Periodic	no	failed (nonpositive definite matrix)		
0.001	Periodic	yes	16.9962	0.1383	1125.2419
0.1	Periodic	yes	failed (nonpositive definite matrix)		

Numerical Test - GPR&SGPR (GPy) Result

Table: GPR&SGPR (GPy) test results: using RBF kernel and bfgs optimizer

Algorithm	m	RMSE		time(s)
		mean	std	
GPR		3.9896	0.0323	2649.0380
SGPR_DTC	1000	3.9964	0.0316	457.3907
SGPR_DTC	500	4.0114	0.0295	222.9858
SGPR_FITC	1000	3.9971	0.0327	554.6843
SGPR_FITC	500	4.01	0.0311	205.7707

Numerical Test - DNN Result

Table: DNN test results: the network width column represents the width of each layer

network width	rmse		time(s)
	mean	std	
single layer			
[64]	4.0507	0.0353	38.5193
[1024]	4.0135	0.0561	65.0261
[5120]	4.025	0.0189	85.3426
[10240]	4.1146	0.0467	111.0864
two layers			
[64, 32]	3.9727	0.0495	47.1162
[1024, 512]	3.9134	0.0545	148.0439

References

Unknown. *UCI public dataset: Combined Cycle Power Plant Data Set*. URL: http://archive.ics.uci.edu/ml/datasets/Combined+Cycle+Power+Plant.