Statistical Techniques in Robotics (16-831, F10) Lecture#19 (Wednesday November 16)

Gaussian Processes

Lecturer: Drew Bagnell *Scribe:Yamuna Krishnamurthy* [1]

1 What problems can be solved by Gaussian Processes?

Supervised learning is the problem of learning input-output mappings from emperical data(input or training dataset). Depending on the characteristics of the output, this problem is known as either regression, for continuous outputs, or classification, when outputs are discrete.

A well known example is the classification of images of handwritten digits. The training set consists of small digitized images, together with a classification from 0, . . . , 9, normally provided by a human. The goal is to learn a mapping from image to classification label, which can then be used on new, unseen images. Supervised learning is an attractive way to attempt to tackle this problem, since it is not easy to specify accurately the characteristics of, say, the handwritten digit 4.

An example of a regression problem in robotics is to learn the inverse dynamics of a robot arm Figure 1(a). Here the task is to map from the state of the arm (given by the positions, velocities and accelerations of the joints) to the corresponding torques on the joints. Such a model can then be used to compute the torques needed to move the arm along a given trajectory. Another example is predictive soil mapping Figure 1(b). Actual soil samples are taken from some regions. These samples can then be used to predict the nature of soil in another region as function of the characteristics of the actual soil samples taken in other regions. Gaussian processes are a supervised learning technique that can be used to solve the problems described above. The following sections define Gaussian Processes in detail, concentrating on its application to regression problems.

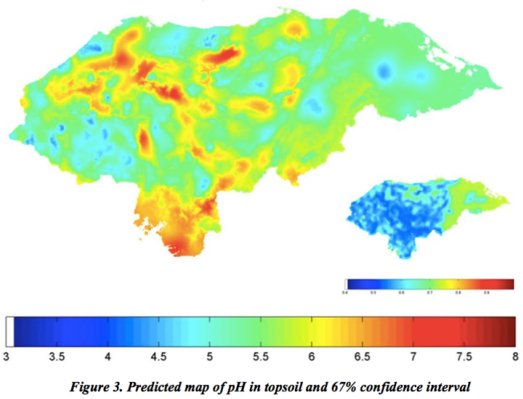
2 How to go about solving them?

In general the procedure is to denote the input as x, and the output (or target) as y. The input is usually represented as a vector x as there are in general many input variables-in the handwritten digit recognition example one may have a 256- dimensional input obtained from a raster scan of a 16×16 image, and in the robot arm example there are three input measurements for each joint in the arm. The target y may either be continuous (as in the regression case) or discrete (as in the classification case). We have a dataset D of n observations, $D = (x_i, y_i)|i = 1, ..., n$.

Given this training data we wish to make predictions of the ouput y for new inputs x^* that we have not seen in the training set. Thus it is clear that the problem at hand is inductive; we need to move from the finite training data D to a function f that makes predictions for all possible input values. To do this we must make assumptions about the characteristics of the underlying function, as otherwise any function which is consistent with the training data would be equally valid.

[1] Some content adapted from previous scribes: Stephane Ross, Joseph Lisee and from Gaussian Processes for Machine Learning by Carl Edward Rasmussen and Christopher K. I. Williams

(a)



(b)

Figure 1: (a) Inverse Kinematics of a Robot Arm [1] and (b) Predictive Soil Modeling [2].

This is where the Gaussian process comes to our rescue. A Gaussian process is a generalization of the Gaussian probability distribution.

3 Gaussian Process (GP)

A gaussian process can be thought of as a gaussian distribution over functions (thinking of functions as infinitely long vectors containing the value of the function at every input). Formally let the input space be $\mathcal{X}$ and $f : \mathcal{X} \to \mathbb{R}$ be a function from the input space to the reals, then we say f is a gaussian process if for any vector of inputs $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$ such that $x_i \in \mathcal{X}$ for all i, the vector of output $f(\mathbf{x}) = [f(x_1), f(x_2), \ldots, f(x_n)]^T$ is gaussian distributed.

The gaussian process is specified by a mean function $\mu : \mathcal{X} \to \mathbb{R}$, such that $\mu(x)$ is the mean of $f(x)$ and a covariance/kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that $k(x_i, x_j)$ is the covariance between $f(x_i)$ and $f(x_j)$. We say $f \sim GP(\mu, k)$ if for any $x_1, x_2, \ldots x_n \in \mathcal{X}$, $[f(x_1), f(x_2), \ldots, f(x_n)]^T$ is gaussian distributed with mean $[\mu(x_1), \mu(x_2), \ldots, \mu(x_n)]^T$ and $n \times n$ covariance/kernel matrix $K_{\mathbf{xx}}$:

$$K_{\mathbf{xx}} = \begin{bmatrix} k(x_1, x_1) & k(x_1, x_2) & \ldots & k(x_1, x_n) \\ k(x_2, x_1) & k(x_2, x_2) & \ldots & k(x_2, x_n) \\ \ldots & \ldots & \ldots & \ldots \\ k(x_n, x_1) & k(x_n, x_2) & \ldots & k(x_n, x_n) \end{bmatrix}$$

The kernel function must have the following properties

- Be symmetric.That is $k(x_i, x_j) = k(x_j, x_i)$
- Be positive definite. That is kernel matrix $K_{\mathbf{xx}}$ induced by k for any set of input is a positive definite matrix.

Example of some kernel functions are given below:

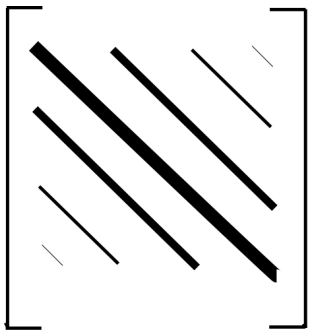


Figure 2: Magnitude of values in covariance matrix, $K_{\mathbf{xx}}$, decrease as the points are further apart.

- Squared Exponential Kernel (Gaussian/RBF): $k(x_i, x_j) = \exp(\frac{-(x_i - x_j)^2}{2\gamma^2})$ where γ is the length scale of the kernel.
- Laplace Kernel: $k(x_i, x_j) = \exp(\frac{-|x_i - x_j|}{\gamma})$.
- Indicator Kernel: $k(x_i, x_j) = I(x_i = x_j)$, where I is the indicator function.
- Linear Kernel: $k(x_i, x_j) = x_i^T x_j$.

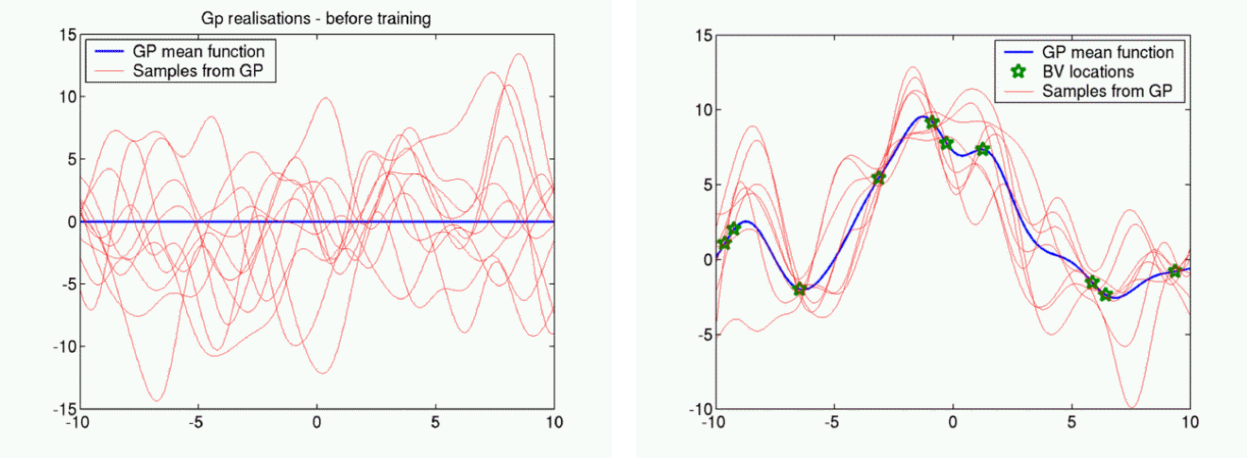
More complicated kernels can be constructed by adding known kernel functions together, as the sum of 2 kernel functions is also a kernel function.

A gaussian process is a random stochastic process where correlation is introduced between neighboring samples (think of a stochastic process as a sequence of random variables). The covariance matrix $K_{\mathbf{xx}}$ has larger values, for points that are closer to each other, and smaller values for points further apart. This can be illustrated in Figure 2. The thicker the line the larger the values. This is because the points are correlated by the difference in their means and their variances. If they are highly correlated, then their means are almost same and the covariance is high.

4 Inference

Gaussian Processes are useful as priors over functions for doing non-linear regression. In Figure 3(a) we see a number of sample functions drawn at random from the prior distribution over functions specified by a particular GP which favours smooth functions. This prior is taken to represent our prior beliefs over the kinds of functions we expect to observe, before seeing any data. In the absence of knowledge to the contrary we have assumed that the average value over the sample functions at each x is zero. Although the specific random functions drawn in Figure 3(a) do not have a mean of zero, the mean of $f(x)$ values for any fixed x would become zero, independent of x as we keep drawing more functions. At any value of x we can also characterize the variability of the sample functions by computing the variance at that point.

Now given a set of observed inputs and corresponding output values $(x_1, f(x_1)), (x_2, f(x_2)), \ldots, (x_n, f(x_n))$, and gaussian process prior on f, $f \sim GP(\mu, k)$, we would like to compute the posterior over the value $f(x^*)$ at any query input x^*. Figure 3(b) illustrates the posterior mean and sample functions from the posterior. We can observe from this that the sample functions from the posterior pass



(a)prior. (b)posterior.

Figure 3: (a) Samples from a zero-mean GP prior and (b) samples from the posterior after a few observations.

close to the observed values but vary a lot in region where there is no observation. So in affect the uncertainty is reduced near the observed values.

4.1 Computing the Posterior

The posterior can be derived similarly to how the update equations for the Kalman filter was derived. First we will find what is the joint distribution of $[f(x^*), f(x_1), f(x_2), \ldots, f(x_n)]^T$, and then use the conditioning rule for gaussian to compute the conditional distribution of $f(x^*)|f(x_1), \ldots, f(x_n)$.

Assume for now that the prior mean function $\mu = 0$. Then the joint distribution of $[f(x^*), f(x_1), f(x_2), \ldots, f(x_n)]^T$ is gaussian:

$$\begin{bmatrix} f(x^*) \\ f(x_1) \\ \ldots \\ f(x_n) \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \\ \ldots \\ 0 \end{bmatrix}, \begin{bmatrix} k(x^*, x^*) & k(x^*, \mathbf{x})^T \\ k(x^*, \mathbf{x}) & K_{\mathbf{xx}} \end{bmatrix} \right)$$

where

$$k(x^*, \mathbf{x}) = \begin{bmatrix} k(x^*, x_1) \\ k(x^*, x_2) \\ \ldots \\ k(x^*, x_n) \end{bmatrix}$$

Now using the conditioning rule we obtained that the posterior for $f(x^*)$ is gaussian:

$$f(x^*)|f(\mathbf{x}) \sim N(\quad k(x^*, \mathbf{x})^T K_{\mathbf{xx}}^{-1} f(\mathbf{x}) \quad , \quad k(x^*, x^*) + k(x^*, \mathbf{x})^T K_{\mathbf{xx}}^{-1} k(x^*, \mathbf{x}) \quad)$$

Notice that the posterior mean $\mathbb{E}(f(x^*)|f(\mathbf{x}))$ can be represented as a linear combination of the kernel function values:

$$\mathbb{E}(f(x^*)|f(\mathbf{x})) = \sum_{i=1}^{n} \alpha_i k(x^*, x_i)$$

for $\alpha = K_{\mathbf{xx}}^{-1} f(\mathbf{x})$. This means we can compute the mean without explicitly inverting K, by solving $K\alpha = f(\mathbf{x})$ instead. Similarly, it can also be represented as a linear combination of the observed function values:

$$\mathbb{E}(f(x^*)|f(\mathbf{x})) = \sum_{i=1}^{n} \beta_i f(x_i)$$

for $\beta = k(x^*, \mathbf{x})^T K^{-1}$.

4.2 Non-zero mean prior

If the prior mean function is non-zero, we can still use the previous derivation by noting that if $f \sim GP(\mu, k)$, then the function $f' = f - \mu$ is a zero-mean gaussian process $f' \sim GP(0, k)$. Hence if we have observations from the values of f, we can subtract the prior mean function values to get observations of f', do the inference on f', and finally once we obtain the posterior on $f'(x^*)$ we can simply add back the prior mean $\mu(x^*)$ to the posterior mean to obtain the posterior on f.

4.3 Noise in observed output values

If instead of having noise-free observation of f, we observe $y(x) = f(x) + \delta$, where $\delta \sim N(0, \sigma^2)$ is some zero-mean gaussian noise, then the joint distribution of $[f(x^*), y(x_1), \ldots y(x_n)]^T$ is also gaussian so that we can apply a similar derivation to compute the posterior of $f(x^*)$. More specifically if the prior mean function $\mu = 0$, we have that:

$$\begin{bmatrix} f(x^*) \\ y(x_1) \\ \cdots \\ y(x_n) \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \\ \cdots \\ 0 \end{bmatrix}, \begin{bmatrix} k(x^*, x^*) + \sigma^2 & k(x^*, \mathbf{x})^T \\ k(x^*, \mathbf{x}) & K_{\mathbf{xx}} + \sigma^2 I \end{bmatrix} \right)$$

The only difference with respect to the noise-free case is that the covariance matrix of the joint now has an extra σ^2 term on its diagonal. This is because the noise is independent among different observations and also independent of f (so no covariance between noise terms, and between f and δ). So we obtain that the posterior on $f(x^*)$ is:

$$f(x^*)|y(\mathbf{x}) \sim N(\; k(x^*, \mathbf{x})^T (K_{\mathbf{xx}} + \sigma^2 I)^{-1} y(\mathbf{x}) \; , \; k(x^*, x^*) + \sigma^2 + k(x^*, \mathbf{x})^T (K_{\mathbf{xx}} + \sigma^2 I)^{-1} k(x^*, \mathbf{x}) \;)$$

4.4 Choosing Kernel Length Scale and Noise Variance Parameters

We can use the data to fit the kernel lenght scale (γ) and noise variance (σ^2) parameters by choosing the parameters that maximizes the log likelihood of the observed data. Assuming a gaussian kernel, then we obtain the most likely parameters γ and σ by solving:

$$\max_{\gamma,\sigma}[\log P(y(\mathbf{x})|\gamma,\sigma)] = \max_{\gamma,\sigma}[-\frac{1}{2}f(\mathbf{x})^T K_{\mathbf{xx}} f(\mathbf{x}) - \frac{1}{2}\log(\det(K_{\mathbf{xx}} + \sigma^2 I)) - \frac{1}{2}\log(2\pi)]$$

Here the determinant will be small when $K_{\mathbf{xx}}$ is almost diagonal, so this maximization will favor smoother kernel (larger γ).

Additionally σ^2 can be chosen to have a higher value to prevent overfitting, as a larger σ means each observation is less informative.

4.5 Computational Complexity

One drawback of the Gaussian Process is that it scales very badly with the number of observations N. Solving for the coefficients α defining the mean function requires $O(N^3)$ computations. Note that bayesian linear regression, which can be seen as a special case of GP with the linear kernel, has complexity of only $O(d^3)$ to find the mean weight vector, for d the dimension of the input space $\mathcal{X}$. Finally to make a prediction at any point, Gaussian Process requires $O(N\hat{d})$ (where $\hat{d}$ is the complexity of evaluating the kernel) while BLR only requires $O(d)$ computations.

References

[1] Botond Bocsi, Duy Nguyen-Tuong, Lehel Csat, Bernhard Schlkopf, Jan Peters, "Learning inverse kinematics with structured prediction," in *IROS 2011: 698-703*

[2] Juan Pablo Gonzalez, Simon Cook, Thomas Oberthur, Andrew Jarvis, J. Andrew (Drew) Bagnell, and M Bernardine Dias, "Creating Low-Cost Soil Maps for Tropical Agriculture using Gaussian Processes," in *Workshop on AI in ICT for Development (ICTD) at the Twentieth International Joint Conference on Artificial Intelligence (IJCAI 2007), January, 2007.*