Final Project 4조_
Dropout & BNN

정예준, 유채원, 이상윤, 이창현, 한지희

진행상황 소개

이론

1. Dropout : A simple way to prevent neural network overfitting
2. Dropout as a Bayesian Approximation
3. Dropout as a Bayesian Approximation : Appendix

Code: MNIST Classification Model을 Dropout CNN from scratch로 구현

1. Torch로 먼저 구현
2. From Scratch로 구현

진행상황 소개

이론

1. Dropout : A simple way to prevent neural network overfitting
2. Dropout as a Bayesian Approximation
3. Dropout as a Bayesian Approximation : Appendix

Code: MNIST Classification Model을 Dropout CNN from scratch로 구현

1. Torch로 먼저 구현
2. From Scratch로 구현

1

Deep Gaussian Process

Deep Gaussian Process

Deep Gaussian Process란?

우리가 세션에서 여러 번 다뤘던 Gaussian Process (GP)의 확장으로,
여러 개의 GP를 계층적으로 쌓아 더 복잡한 데이터를 모델링 하는 것이다.
-> 마치 딥러닝에서 여러 개의 layer를 쌓는 것과 비슷한 느낌.
핵심은, 이전 GP의 출력을 다음 GP의 입력으로 사용하는 것이다.

Deep Gaussian Process (A.C. Damianou, N.D. Lawerence, 2012)
논문을 간단히 살펴보며 Deep Gaussian Process를 이해해보자.

먼저, Deep Gaussian Process에 대해 이해하기 전에, 이전의
Standard Gaussian Process의 구조를 간단히 복습해보자.

input-output pair (X, Y)가 주어졌을 때, 우리는 다음 모델을
통해 latent function $f(x)$를 estimate 하려 한다.

$$y_n = f(x_n) + \epsilon_n, \quad \epsilon_n \sim \mathcal{N}(0, \sigma_\epsilon^2)$$
$$\text{where } f(x) \sim \mathcal{GP}(0, k(x, x'))$$

즉, 각 y_n은 $f(x_n)$에 independent Gaussian Noise를 더해 만들어진
다.
이때, marginal likelihood는 다음과 같이 주어진다.

$$p(Y|X) = \mathcal{N}(Y|0, K_{NN} + \sigma_\epsilon^2 I)$$

여기서 $K_{NN} = k(X, X)$ 는 Covariance Matrix이다.

Deep Gaussian Process

Deep Gaussian Process는 latent variable들이 여러 layer 쌓여 만들어진다. Deep GP에는 다음 세 가지 type의 node들이 있다.

1. Leaf nodes $Y \in \mathbb{R}^{N \times D}$ (observed)
2. Intermediate latent spaces $X_h \in \mathbb{R}^{N \times Q_h}$ for $h = 1, \ldots, H-1$
3. Parent latent node $Z = X_H \in \mathbb{R}^{N \times Q_Z}$

Deep GP에서는, intermediate Node X_h가 이전 layer의 output 이면서 동시에 다음 layer의 input으로 사용된다.
예를 들어, 2 개의 Hidden layer를 가진 Deep GP 모델은 아래와 같이 표현될 수 있다.

$$y_{nd} = f_Y^d(x_n) + \epsilon_{nd}, \quad x_{nq} = f_X^q(z_n) + \epsilon_{nq}$$
$$\text{with } f_Y \sim \mathcal{GP}(0, k_Y(X, X)) \text{ and } f_X \sim \mathcal{GP}(0, k_X(Z, Z))$$

Deep GP는 Gaussian Process가 중첩된 구조이기 때문에, Layer가 많아질수록 posterior를 직접적으로 추론하는 것은 사실상 불가능하다. 따라서, 우리는 Variational Inference 및 Bayesian approach를 이용할 것이다.

Bayesian approach를 사용하기 위한 첫 단계는 Gaussian Process 들의 covariance function으로 다음과 같은 automatic relevance determination (ARD) covariance function을 사용하는 것이다.

$$k(x_i, x_j) = \sigma_{ard}^2 \exp\left(-\frac{1}{2}\sum_{q=1}^{Q} w_q (x_{i,q} - x_{j,q})^2\right)$$

이 covariance function은 각각의 latent dimension에 대해 각각 다른 weight w_q를 가정한다.

Deep Gaussian Process

이제, deep GP에서의 Bayesian Training을 살펴보자.

Bayesian Training은 다음의 model evidence를 Optimize 해야 한다.

$$\log p(Y) = \log \int p(Y|X)p(X|Z)p(Z)\, dX\, dZ$$

만약 Prior distribution을 가정할 수 있다면, parent latent node (= 첫 layer)의 Prior가 latent space를 적절하게 constrain해줄 수 있다.

그러나, 이 적분은 계산이 불가능하다. (X, Z가 GP를 거치며 nonlinear 해지기 때문)

따라서, Jensen's Inequality를 이용하여 Variational Lower bound $\mathcal{F}_v \leq \log p(Y)$ 도해내고, 이때

$$\mathcal{F}_v = \int_{\mathbf{X},\mathbf{Z},\mathbf{F}^Y,\mathbf{F}^X} \mathcal{Q} \log \frac{p(\mathbf{Y},\mathbf{F}^Y,\mathbf{F}^X,\mathbf{X},\mathbf{Z})}{\mathcal{Q}}$$

이때, 분자의 Joint distribution은 다음과 같이 전개된다.

$$p(Y, F_Y, F_X, X, Z) = p(Y|F_Y)p(F_Y|X)p(X|F_X)p(F_X|Z)p(Z)$$

Intractable한 Term들을 다루기 위해, pseudo-input $\tilde{X} \in \mathbb{R}^{K\times Q}$ $\tilde{Z} \in \mathbb{R}^{K\times Q_Z}$가.

이때, 이들은 Function Value $U_Y \in R^{K\times D}, U_X \in R^{K\times Q}$ 대응한다.

이제, 위의 Joint Distribution을 pseudo-input들을 도입하여 다음과 같이 바꿔 쓸 수 있다.

$$p(Y, F_Y, F_X, X, Z, U_Y, U_X, \tilde{X}, \tilde{Z}) =$$
$$p(Y|F_Y)p(F_Y|U_Y, X)p(U_Y|\tilde{X})p(X|F_X)p(F_X|U_X, Z)p(U_X|\tilde{X})p(Z)$$

Deep Gaussian Process

이제, Variational Distribution Q를 다음과 같이 정의하자.

$$Q = p(F_Y|U_Y, X)q(U_Y)q(X)p(F_X|U_X, Z)q(U_X)q(Z)$$

이때, $q(X) = \prod_{q=1}^{Q} \mathcal{N}(\mu_X^q, S_X^q), \quad q(Z) = \prod_{q=1}^{Q_Z} \mathcal{N}(\mu_Z^q, S_Z^q)$

$q(U_Y)$와 $q(U_X)$는 Variational Distribution에서 free한 분포를 선택해야 한다.

이제, Variational Lower Bound는 다음과 같이 정리된다.

$$\mathcal{F}_v = \int Q \log \frac{p(Y|F_Y)p(U_Y)p(X|F_X)p(U_X)p(Z)}{Q_0} \, dX \, dZ \, dF_Y \, dF_X$$

where $Q_0 = q(U_Y)q(X)q(U_X)q(Z)$

Log를 풀고 같은 변수들끼리 모아주면 최종적으로 Variational Lower Bound는 아래와 같이 표현할 수 있다.

$$\mathcal{F}_v = g_Y + r_X + H_q(X) - \mathrm{KL}(q(Z)\|p(Z))$$

이때,

$$g_Y = \langle \log p(Y|F_Y) + \log p(U_Y) \rangle_{p(F_Y|U_Y,X)q(U_Y)q(X)}$$
$$r_X = \langle \log p(X|F_X) + \log p(U_X) \rangle_{p(F_X|U_X,Z)q(U_X)q(X)q(Z)}$$

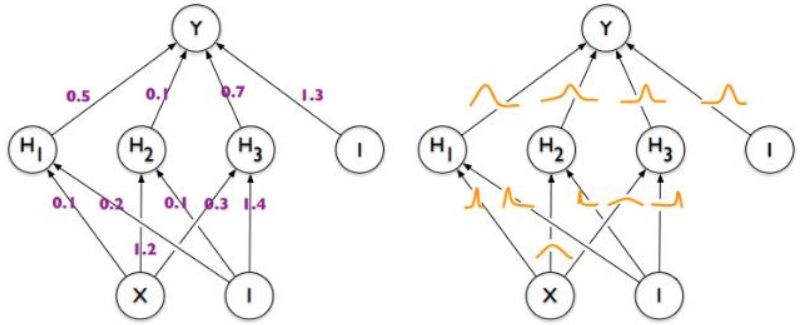
또, H는 entropy이고, KL은 KL Divergence term이다.

여기에 사용된 term들은 모두 우리가 알고 있는 Gaussian term이기 때문에, 계산이 가능하게 되어 Variational Inference를 통해 Deep GP의 Posterior를 구할 수 있게 된다.

2
Approximation for Regression

Remind

Remind



Weight를 Unknown fixed constant로 보던 기존의 Deterministic한 모델과 달리, BNN에서는 weight에 어떠한 확률 분포를 가정한다.
즉, 우리는 주어진 데이터와 prior에 대한 가정을 통해 함수의 posterior를 찾고, 이를 이용해 모델을 구축한다.

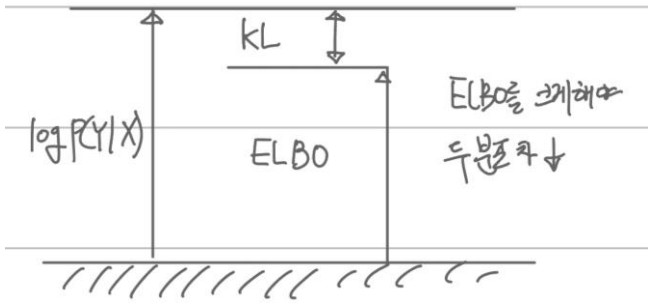
$$p(\boldsymbol{\omega}|\mathbf{x},\mathbf{y}) \propto p(\mathbf{y}|\mathbf{x},\boldsymbol{\omega})p(\boldsymbol{\omega})$$

Posterior을 이용해 marginalization하여 다음의 predictive distribution을 구할 수 있다.

$$p(\mathbf{y}^*|\mathbf{x}^*,\mathbf{X},\mathbf{Y}) = \int p(\mathbf{y}^*|\mathbf{x}^*,\boldsymbol{\omega})p(\boldsymbol{\omega}|\mathbf{x},\mathbf{y})d\boldsymbol{\omega}$$

하지만 실제 상황에서 Bayes rule을 이용한 posterior 계산이 어렵거나, 불가능한 경우가 있으므로 Variational Inference를 활용한다.
즉, Posterior와 비슷한 분포를 갖는 Variational 분포인 q분포를 정의하고, 둘을 근사시키는 방법으로 posterior를 추론한다.

이때 posterior와 q의 KL-분산을 최소화시키는 방법을 통해 q를 posterior에 근사시킨다.
그러나, posterior를 모르기 때문에 KL-분산을 직접적으로 구할 수 없고, ELBO(Evidence Lowe Bound)를 사용하게 된다.
Marginal log likelihood는 ELBO와 KL-분산의 합으로 구성되기 때문에, KL-분산을 최소화하는 문제는 ELBO를 최대화하는 문제와 같아진다.



Remind

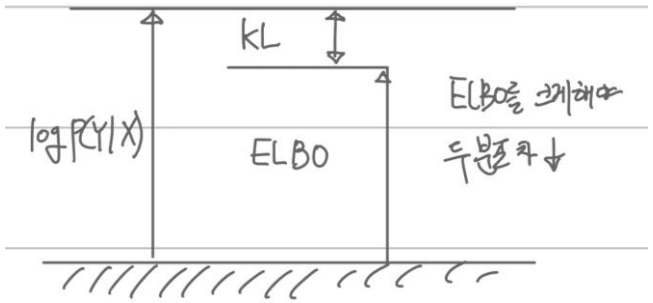
Proof. Minimize KL ⇔ Maximize ELBO

$$
\begin{aligned}
\mathrm{KL}(q(\boldsymbol{\omega})||p(\boldsymbol{\omega}|\mathbf{x},\mathbf{y})) &= \int q(\boldsymbol{\omega}) \log \tfrac{q(\boldsymbol{\omega})}{p(\boldsymbol{\omega}|\mathbf{x},\mathbf{y})} d\boldsymbol{\omega} \\
&= \int q(\boldsymbol{\omega}) \log q(\boldsymbol{\omega}) d\boldsymbol{\omega} - \int q(\boldsymbol{\omega}) \log p(\boldsymbol{\omega}|\mathbf{x},\mathbf{y}) d\boldsymbol{\omega} \\
&= \int q(\boldsymbol{\omega}) \log q(\boldsymbol{\omega}) d\boldsymbol{\omega} - \int q(\boldsymbol{\omega}) \log \tfrac{p(\mathbf{y}|\mathbf{x},\boldsymbol{\omega})p(\boldsymbol{\omega})}{p(\mathbf{y}|\mathbf{x})} d\boldsymbol{\omega} \\
&= \int q(\boldsymbol{\omega}) \log q(\boldsymbol{\omega}) d\boldsymbol{\omega} - \int q(\boldsymbol{\omega}) \log p(\mathbf{y}|\mathbf{x},\boldsymbol{\omega}) d\boldsymbol{\omega} - \int q(\boldsymbol{\omega}) \log p(\boldsymbol{\omega}) d\boldsymbol{\omega} + \int q(\boldsymbol{\omega}) \log p(\mathbf{y}|\mathbf{x}) d\boldsymbol{\omega} \\
&= \int q(\boldsymbol{\omega}) \log \tfrac{q(\boldsymbol{\omega})}{p(\boldsymbol{\omega})} d\boldsymbol{\omega} - \int q(\boldsymbol{\omega}) \log p(\mathbf{y}|\mathbf{x},\boldsymbol{\omega}) d\boldsymbol{\omega} + \int q(\boldsymbol{\omega}) \log p(\mathbf{y}|\mathbf{x}) d\boldsymbol{\omega} \\
&= \int q(\boldsymbol{\omega}) \log \tfrac{q(\boldsymbol{\omega})}{p(\boldsymbol{\omega})} d\boldsymbol{\omega} - \int q(\boldsymbol{\omega}) \log p(\mathbf{y}|\mathbf{x},\boldsymbol{\omega}) d\boldsymbol{\omega} + \log p(\mathbf{y}|\mathbf{x})
\end{aligned}
$$

$$
\mathrm{KL}(q(\boldsymbol{\omega})||p(\boldsymbol{\omega}|\mathbf{x},\mathbf{y})) = \mathrm{KL}(q(\boldsymbol{\omega})||p(\boldsymbol{\omega})) - \int q(\boldsymbol{\omega}) \log p(\mathbf{y}|\mathbf{x},\boldsymbol{\omega}) d\boldsymbol{\omega} + \log p(\mathbf{y}|\mathbf{x})
$$

$$
\log p(\mathbf{y}|\mathbf{x}) = \underbrace{\int q(\boldsymbol{\omega}) \log p(\mathbf{y}|\mathbf{x},\boldsymbol{\omega}) d\boldsymbol{\omega} - \mathrm{KL}(q(\boldsymbol{\omega})||p(\boldsymbol{\omega}))}_{\mathcal{L}_{\mathrm{VI}}} + \underbrace{\mathrm{KL}(q(\boldsymbol{\omega})||p(\boldsymbol{\omega}|\mathbf{x},\mathbf{y}))}_{\text{KL-분산} \geq 0}
$$

$\mathcal{L}_{\mathrm{VI}}$: ELBO(Evidence Lower Bound)

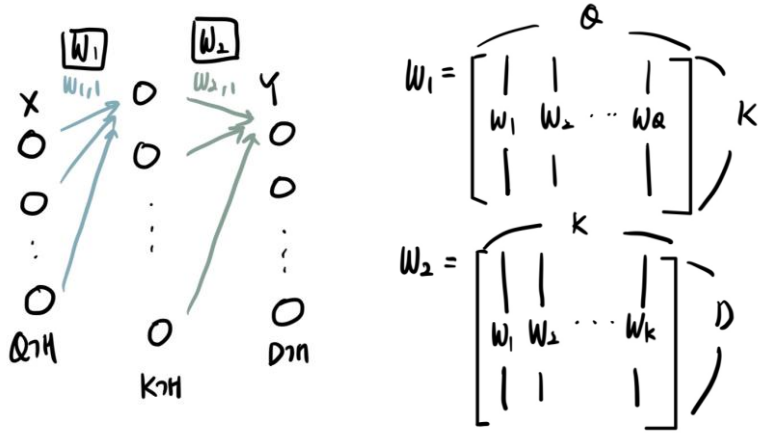


Gaussian Process approximation

우리는 간단하게 single layer 상황을 가정한다.

<Guassian Process>

$$\mathbf{F} \mid \mathbf{X}, \mathbf{W}_1, \mathbf{b} \sim \mathcal{N}(\mathbf{0}, \widehat{\mathbf{K}}(\mathbf{X}, \mathbf{X}))$$
$$\mathbf{Y} \mid \mathbf{F} \sim \mathcal{N}(\mathbf{F}, \tau^{-1}\mathbf{I}_N),$$
$$\mathbf{w}_k \sim p(\mathbf{w}),\ b_k \sim p(b),$$
$$\mathbf{W}_1 = [\mathbf{w}_k]_{k=1}^K, \mathbf{b} = [b_k]_{k=1}^K$$



Neural Network으로 해석하기 위하여 K(X,X)를 다음과 같이 정의해준다.

$$\mathbf{K}(\mathbf{x}, \mathbf{y}) = \int p(\mathbf{w})p(b)\sigma(\mathbf{w}^T\mathbf{x} + b)\sigma(\mathbf{w}^T\mathbf{y} + b)\mathrm{d}\mathbf{w}\mathrm{d}b \xrightarrow[\text{MCMC Approximate}]{} \widehat{\mathbf{K}}(\mathbf{x}, \mathbf{y}) = \frac{1}{K}\sum_{k=1}^{K}\sigma(\mathbf{w}_k^T\mathbf{x} + b_k)\sigma(\mathbf{w}_k^T\mathbf{y} + b_k)$$

Predictive Distribution: 다음과 같이 바뀔 수 있도록 W_2를 설정해보자

$$p(\mathbf{Y}|\mathbf{X}) = \int p(\mathbf{Y}|\mathbf{F})p(\mathbf{F}|\mathbf{W}_1, \mathbf{b}, \mathbf{X})p(\mathbf{W}_1)p(\mathbf{b}) \longrightarrow p(\mathbf{Y}|\mathbf{X}) = \int p(\mathbf{Y}|\mathbf{X}, \mathbf{W}_1, \mathbf{W}_2, \mathbf{b})p(\mathbf{W}_1)p(\mathbf{W}_2)p(\mathbf{b})$$

Gaussian Process approximation

Gaussian Process의 Covariance Function은 Kernel Function의 형태를 지닌다.

$\phi(\mathbf{x}, \mathbf{W}_1, \mathbf{b}) = \sqrt{\frac{1}{K}}\sigma(\mathbf{W}_1^T\mathbf{x} + \mathbf{b})$ 로 정의하면,

$$\widehat{\mathbf{K}}(\mathbf{X}, \mathbf{X}) = \Phi\Phi^T$$

$$p(\mathbf{Y}|\mathbf{X}) = \int p(\mathbf{Y}|\mathbf{F})p(\mathbf{F}|\mathbf{W}_1, \mathbf{b}, \mathbf{X})p(\mathbf{W}_1)p(\mathbf{b})$$

$$\mathbf{F} \mid \mathbf{X}, \mathbf{W}_1, \mathbf{b} \sim \mathcal{N}(\mathbf{0}, \widehat{\mathbf{K}}(\mathbf{X}, \mathbf{X}))$$
$$\mathbf{Y} \mid \mathbf{F} \sim \mathcal{N}(\mathbf{F}, \tau^{-1}\mathbf{I}_N),$$

$$p(\mathbf{Y}|\mathbf{X}) = \int \mathcal{N}(\mathbf{Y}; \mathbf{0}, \Phi\Phi^T + \tau^{-1}\mathbf{I}_N)p(\mathbf{W}_1)p(\mathbf{b})\mathrm{d}\mathbf{W}_1\mathrm{d}\mathbf{b},$$

$\mathbf{w}_d \sim \mathcal{N}(0, \mathbf{I}_K)$ 로 정의

$$\mathcal{N}(\mathbf{y}_d; 0, \Phi\Phi^T + \tau^{-1}\mathbf{I}_N) = \int \mathcal{N}(\mathbf{y}_d; \Phi\mathbf{w}_d, \tau^{-1}\mathbf{I}_N)\mathcal{N}(\mathbf{w}_d; 0, \mathbf{I}_K)\mathrm{d}\mathbf{w}_d.$$ 즉, w_d에 대한 marginalization

$$p(\mathbf{Y}|\mathbf{X}) = \int p(\mathbf{Y}|\mathbf{X}, \mathbf{W}_1, \mathbf{W}_2, \mathbf{b})p(\mathbf{W}_1)p(\mathbf{W}_2)p(\mathbf{b})$$

Predictive Distribution을 X, W_1, W_2, b, Y로 표현할 수 있다.

Gaussian Process approximation

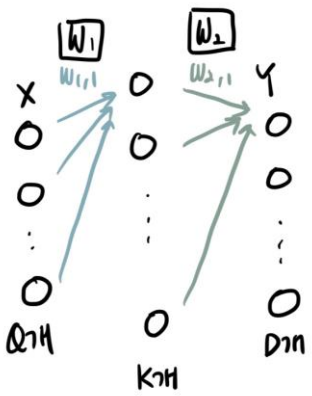
W_1, W_2, b에 대한 Posterior를 구해야 한다.

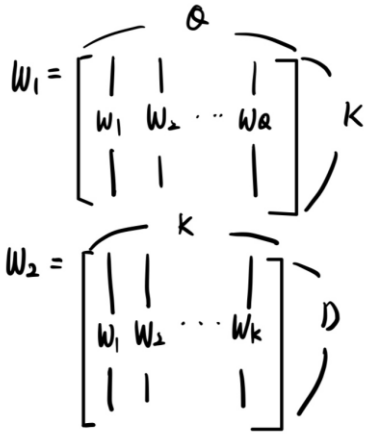
➔ Variational Distribution 이용

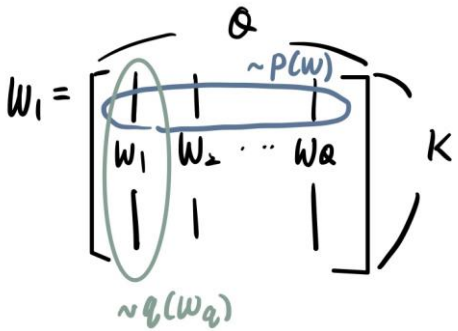
W_1, W_2에 대한 variational distribution으로 Gaussian Mixture Distribution을 사용한다.

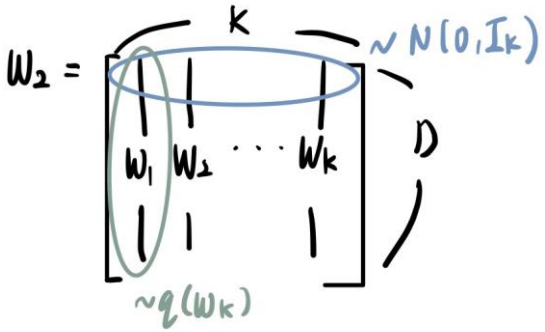
$$q(\mathbf{W}_1) = \prod_{q=1}^{Q} q(\mathbf{w}_q), \qquad q(\mathbf{W}_2) = \prod_{k=1}^{K} q(\mathbf{w}_k), \qquad q(\mathbf{b}) = \mathcal{N}(\mathbf{m}, \sigma^2 \mathbf{I}_K)$$

$$q(\mathbf{w}_q) = p_1 \mathcal{N}(\mathbf{m}_q, \sigma^2 \mathbf{I}_K) + (1 - p_1)\mathcal{N}(0, \sigma^2 \mathbf{I}_K) \qquad q(\mathbf{w}_k) = p_2 \mathcal{N}(\mathbf{m}_k, \sigma^2 \mathbf{I}_D) + (1 - p_2)\mathcal{N}(0, \sigma^2 \mathbf{I}_D)$$









Evaluating the Log Evidence Lower Bound for Regression

ELBO를 다음과 같이 G.P를 활용해 표현해준다.

$$\mathcal{L}_{\text{VI}} := \int q(\boldsymbol{\omega}) \log p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\omega}) d\boldsymbol{\omega} - \text{KL}(q(\boldsymbol{\omega})||p(\boldsymbol{\omega}))$$

$$\mathcal{L}_{\text{GP-VI}} := \int q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) \log p(\mathbf{Y}|\mathbf{X}, \mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) - \text{KL}(q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b})||p(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}))$$ where the integration is with respect to $\mathbf{W}_1$, $\mathbf{W}_2$, and $\mathbf{b}$.

(1) First term of ELBO

$$\begin{aligned}
\log p(\mathbf{Y}|\mathbf{X}, \mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) &= \sum_{d=1}^{D} \log \mathcal{N}(\mathbf{y}_d; \boldsymbol{\Phi}\mathbf{w}_d, \tau^{-1}\mathbf{I}_N) \\
&= -\frac{ND}{2}\log(2\pi) + \frac{ND}{2}\log(\tau) - \sum_{d=1}^{D}\frac{\tau}{2}||\mathbf{y}_d - \boldsymbol{\Phi}\mathbf{w}_d||^2 \\
&= -\frac{ND}{2}\log(2\pi) + \frac{ND}{2}\log(\tau) - \sum_{n=1}^{N}\frac{\tau}{2}||\mathbf{y}_n - \hat{\mathbf{y}}_n||^2 \\
&= \sum_{n=1}^{N} \log \mathcal{N}(\mathbf{y}_n; \boldsymbol{\phi}(\mathbf{x}_n, \mathbf{W}_1, \mathbf{b})\mathbf{W}_2, \tau^{-1}\mathbf{I}_D)
\end{aligned}$$

$\log P(Y|X, W_1, W_2, b) = \sum_{d=1}^{D} \log \mathcal{N}(y_d; \Phi W_d, \tau^{-1} I_N)$ $\quad \left|\begin{matrix} \tau^{-1} & & \\ & \tau^{-1} & \\ & & \ddots \end{matrix}\right|$

이때 $\mathcal{N}(y_d; \Phi W_d, \tau^{-1} I_N) = \frac{1}{(2\pi)^{N/2}|\tau^{-1}I_N|^{1/2}} \exp\left[-\frac{1}{2}(y_d - \Phi W_d)^T (\tau^{-1}I_N)^{-1}(y_d - \Phi W_d)\right]$, $|\tau^{-1}I_N| = \tau^{-N}$

$\log \mathcal{N}(y_d; \Phi W_d, \tau^{-1} I_N) = -\frac{N}{2}\log(2\pi) + \frac{N}{2}\log(\tau) + \left[-\frac{\tau}{2}(y_d - \Phi W_d)^T (y_d - \Phi W_d)\right]$

이를 D번 더한 결과는 $= -\frac{ND}{2}\log(2\pi) + \frac{ND}{2}\log(\tau) - \frac{\tau}{2}\sum_{d=1}^{D}\|y_d - \Phi W_d\|^2$ 이고

$\hat{y} = \Phi W_d$ 이므로, $\sum_{d=1}^{D} \frac{\tau}{2}\|y_d - \hat{y}_d\|^2$ 이다.

현재 $\hat{y}$는 N개의 변수, D개의 관측치를 갖는데, 이를 반대로 N번관측치, D개의 변수가 생성하면

$\sum_{d=1}^{D} \frac{\tau}{2}\|y_d - \hat{y}_d\|^2 = \sum_{n=1}^{N} \frac{\tau}{2}\|y_n - \hat{y}_n\|^2$

또한 $\hat{y}_n = \phi(x_n, W_1, b) W_2 = \sqrt{\frac{1}{K}}\sigma(x_n W_1 + b) W_2$ 이기에

최종적으로 $\log P(Y|X, W_1, W_2, b) = \sum_{n=1}^{N} \log \mathcal{N}(y_n; \phi(x_n, W_1, b) W_2, \tau^{-1} I_D)$ 같음의 행렬 연산법칙.

위 내용을 ELBO에 적용하면 다음과 같다.

$$\mathcal{L}_{\text{GP-VI}} := \sum_{n=1}^{N} \int q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) \log p(\mathbf{y}_n|\mathbf{x}_n, \mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) - \text{KL}(q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b})||p(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}))$$ where the integration is with respect to $\mathbf{W}_1$, $\mathbf{W}_2$, and $\mathbf{b}$.

Evaluating the Log Evidence Lower Bound for Regression

$$\mathcal{L}_{\text{GP-VI}} := \sum_{n=1}^{N} \int q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) \log p(\mathbf{y}_n | \mathbf{x}_n, \mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) - \text{KL}(q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) || p(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}))$$ where the integration is with respect to $\mathbf{W}_1$, $\mathbf{W}_2$, and $\mathbf{b}$.

$\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}$ 을 sampling하여 monte-carlo integration?

하지만 parameter $\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}$ 가 random하므로 이를 구하기 쉽지 않다.

이 문제를 Re-parametrization trick을 이용해 해결할 수 있다. Ex) $x \sim \mathcal{N}(\mu, \sigma)$ ->$x = \mu + \sigma \times \varepsilon, \;\; where \;\; \varepsilon \sim \mathcal{N}(0, 1)$

다음과 같이 parameter $\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}$ 를 stochastic part, deterministic part로 나누어 바꿔준다.

$$\mathbf{W}_1(\mathbf{z}_1, \varepsilon_1) = \mathbf{z}_1(\mathbf{M}_1 + \boldsymbol{\sigma}\varepsilon_1) + (1 - \mathbf{z}_1)\boldsymbol{\sigma}\varepsilon_1$$
$$\mathbf{W}_2(\mathbf{z}_2, \varepsilon_2) = \mathbf{z}_2(\mathbf{M}_2 + \boldsymbol{\sigma}\varepsilon_2) + (1 - \mathbf{z}_2)\boldsymbol{\sigma}\varepsilon_2$$
$$\mathbf{b}(\varepsilon) = \mathbf{m} + \boldsymbol{\sigma}\varepsilon$$

where $q(\varepsilon_1) = \mathcal{N}(\mathbf{0}, \mathbf{I}_{Q \times K}),\ q(\varepsilon_2) = \mathcal{N}(\mathbf{0}, \mathbf{I}_{K \times D}),\ q(\varepsilon) = \mathcal{N}(\mathbf{0}, \mathbf{I}_K)$

$q(\mathbf{z}_{1,q}) = \mathbf{Bernoulli}(p_1)\, for\, q = 1, , , Q \quad q(\mathbf{z}_{2,k}) = \mathbf{Bernoulli}(p_2)\, for\, k = 1, , , K$

ELBO의 first term에 이를 적용하고, 각 $\mathbf{z}$, ε 에 대하여 sampling하여 monte-carlo integration하면 다음의 ELBO를 얻는다.

$$\mathcal{L}_{\text{GP-MC}} := \sum_{n=1}^{N} \log p(\mathbf{y}_n | \mathbf{x}_n, \hat{\mathbf{W}}_1^n, \hat{\mathbf{W}}_2^n, \hat{\mathbf{b}^n}) - \text{KL}(q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) || p(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}))$$

Evaluating the Log Evidence Lower Bound for Regression

$$\mathcal{L}_{\text{GP}-\text{MC}} := \sum_{n=1}^{N} \log p(\mathbf{y}_n|\mathbf{x}_n, \hat{\mathbf{W}}_1^n, \hat{\mathbf{W}}_2^n, \hat{\mathbf{b}^n}) - \text{KL}(q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b})||p(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}))$$

(2) Second term of ELBO

Proposition 1.Gaussian Mixture와 Single Gaussian의 KL-분산 근사식

> $Let\ K, L \in \mathbb{N}\ and\ \pi_i$ is probability vector(mixing coefficient)
>
> $q(\mathbf{x}) = \sum_{i=1}^{L} \pi_i \mathcal{N}(\mathbf{x};\ \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \quad where\ \boldsymbol{\mu}_i \in \mathbb{R}^K,\ \boldsymbol{\Sigma}_i \in \mathbb{R}^{K\times K}$
>
> $p(\mathbf{x}) = \mathcal{N}(\mathbf{0}, \mathbf{I}_k)$
>
> 이때의 KL분산은 다음과 같이 근사 가능하다.
>
> $\text{KL}(q(\mathbf{x})||p(\mathbf{x})) \approx \sum_{i=1}^{L} \frac{\pi_i}{2}(\mathbf{m}_i^T\mathbf{m}_i + tr(\boldsymbol{\Sigma}_i) - K(1+\log 2\pi) - \log|\boldsymbol{\Sigma}_i|) + constant$

Proposition 1. 을 사용하면 ELBO식의 KL-분산이 다음과 같은 approximate된다.

$$\mathbf{KL}(q(\mathbf{W_1})||p(\mathbf{W_1})) \approx QK(\sigma^2 - \log(\sigma^2) - 1) + \frac{p_1}{2}\sum_{q=1}^{Q}(\mathbf{m}_q^T\mathbf{m}_q) + c$$

$$\mathbf{KL}(q(\mathbf{W_2})||p(\mathbf{W_2})) \approx DK(\sigma^2 - \log(\sigma^2) - 1) + \frac{p_2}{2}\sum_{k=1}^{K}(\mathbf{m}_k^T\mathbf{m}_k) + c$$

$$\mathbf{KL}(q(\mathbf{b})||p(\mathbf{b})) = \frac{1}{2}(\mathbf{m}^T\mathbf{m} + K(\sigma^2 - \log(\sigma^2) - 1)) + c$$

where $q(\mathbf{W}_1) = \prod_{q=1}^{Q} q(\mathbf{w}_q) \quad q(\mathbf{w}_q) = p_1\mathcal{N}(\mathbf{m}_q,\ \sigma^2\mathbf{I}_k) + (1-p_1)\mathcal{N}(0,\ \sigma^2\mathbf{I}_k)$

$q(\mathbf{W}_2) = \prod_{k=1}^{K} q(\mathbf{w}_k) \quad q(\mathbf{w}_k) = p_2\mathcal{N}(\mathbf{m}_k,\ \sigma^2\mathbf{I}_D) + (1-p_2)\mathcal{N}(0,\ \sigma^2\mathbf{I}_D)$

$q(\mathbf{b}) = \mathcal{N}(\mathbf{m},\ \sigma^2\mathbf{I}_k)$

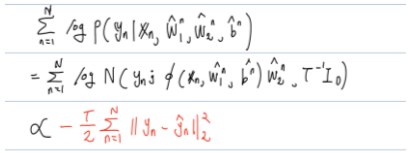
Log Evidence Lower Bound Optimization

$$\mathcal{L}_{\text{GP-MC}} := \sum_{n=1}^{N} \log p(\mathbf{y}_n|\mathbf{x}_n, \hat{\mathbf{W}}_1^n, \hat{\mathbf{W}}_2^n, \hat{\mathbf{b}^n}) - \text{KL}(q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b})||p(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}))$$

The Maximization Objective

$$\mathcal{L}_{\text{GP-MC}} \propto -\frac{\tau}{2}\sum_{n=1}^{N}||\mathbf{y}_n - \hat{\mathbf{y}_n}||_2^2 - \frac{p_1}{2}||\mathbf{M}_1||_2^2 - \frac{p_2}{2}||\mathbf{M}_2||_2^2 - \frac{1}{2}||\mathbf{m}||_2^2$$

첫 번째 term:



두 번째 ~ 네 번째 term: approximate한 KL식에서 σ을 0으로 보내 최적화에 필요한 부분만 남긴다.

$$\hat{\mathbf{W}}_1 \approx \hat{\mathbf{z}_1^n}\mathbf{M}_1 \quad \hat{\mathbf{W}}_2 \approx \hat{\mathbf{z}_2^n}\mathbf{M}_2 \quad \hat{\mathbf{b}}^n \approx \mathbf{m} \qquad \hat{\mathbf{y}}_n \approx \sqrt{\frac{1}{K}}\sigma(\mathbf{x}_n(\hat{\mathbf{z}_1^n}\mathbf{M}_1) + \mathbf{m})(\hat{\mathbf{z}_2^n}\mathbf{M}_2)$$

Scaling objective by $\frac{1}{\tau N}$

$$\mathcal{L}_{\text{GP-MC}} \propto -\frac{1}{2N}\sum_{n=1}^{N}||\mathbf{y}_n - \hat{\mathbf{y}_n}||_2^2 - \frac{p_1}{2\tau N}||\mathbf{M}_1||_2^2 - \frac{p_2}{2\tau N}||\mathbf{M}_2||_2^2 - \frac{1}{2\tau N}||\mathbf{m}||_2^2$$

NN with dropout, L2 Regularization의 LOSS

$$\mathcal{L}_{\text{dropout}} := \frac{1}{2N}\sum_{n=1}^{N}||\mathbf{y}_n - \hat{\mathbf{y}_n}||_2^2 + \lambda_1||\mathbf{W}_1||_2^2 + \lambda_2||\mathbf{W}_2||_2^2 + \lambda_3||\mathbf{b}||_2^2$$

ELBO식을 최대화하는 것이 NN의 LOSS를 최소화하는 것과 같다.

즉, NN with dropout, L2 Regularization을 적용하는 것은 Deep Gaussian Process의 Bayesian Approximation으로 해석 가능하다.

3

Extensions

- Evaluating ELBO for Classification
- Prior Scaling

Evaluating the ELBO for Classification

- 지금까지 회귀 과제에서 사후 분포를 근사한 분포의 ELBO를 최대화하는 것이 드롭아웃과 L2 regularization을 적용한 NN의 손실 함수를 최소화하는 것과 같음을 보였다.
  같은 결론이 분류 과제에서도 성립하는지 확인해보자.

- 회귀 과제에서와 달리, 분류 과제에서는 output인 Y에 softmax 함수를 적용해 얻은 확률로 최종 class를 얻는 과정이 추가된다.

$$
\begin{aligned}
&w_k \sim p(w), \quad b_k \sim p(b), \\
&W_1 = [w_k]k = 1^K, \quad b = [b_k]k = 1^K \\
&\hat{K}(x,y) = \frac{1}{K}\sum_{k=1}^{K}\sigma(w_k^T x + b_k)\sigma(w_k^T y + b_k) \\
&F|X, W_1, b \sim \mathcal{N}(0, \hat{K}(X,X)) \\
&Y|F \sim \mathcal{N}(F, 0 \cdot I_N) \\
&c_n|Y \sim \text{Categorical}\left(\exp(y_{nd})/\left(\sum_{d'}\exp(y_{nd'})\right)\right)
\end{aligned}
$$

- 따라서 분류 과제에서의 predictive model은 다음과 같다. 이 때 $p(\boldsymbol{Y}|\boldsymbol{X})$는 회귀 과제에서 유도한 predictive model의 수식과 동일하다.

$$
\begin{aligned}
p(c|X) &= \int p(c|Y)p(Y|X)dY \\
&= \int p(c|Y)\left(\int p(Y|X, W_1, W_2, b)p(W_1, W_2, b)dW_1 dW_2 db\right)dY
\end{aligned}
$$

- 회귀 과제에서 사용했던 Variational dist'n과 Variational parameter을 그대로 사용해 ELBO를 유도하면 다음과 같다.

$$
\begin{aligned}
\mathcal{L}_{\text{GP-VI}} := &\int p(Y|X, W_1, W_2, b)q(W_1, W_2, b)\log p(c|Y)dW_1 dW_2 db dY \\
&- \text{KL}(q(W_1, W_2, b)||p(W_1, W_2, b))
\end{aligned}
$$

- 회귀에서와 동일하게, 위 식의 (1) 첫 번째 항에 대해 re-parametrization trick과 MC integration을 적용하고 (2) 두 번째 항의 KL-divergence를 근사해 maximization function을 얻는다. 이 식을 (3) $\frac{1}{N}$으로 scaling 하면 아래와 같은 식을 얻을 수 있다.

$$
\mathcal{L}_{\text{GP-MC}} \propto \frac{1}{N}\sum_{n=1}^{N}\hat{p}_{n,c_n} - \frac{p_1}{2N}||M_1||2^2 - \frac{p_2}{2N}||M_2||2^2 - \frac{1}{2N}||m||2^2
$$

ELBO vs. Loss Function for Classification

- 한 편, 분류 과제에서는 NN의 error function으로 softmax loss $-\frac{1}{N}\sum_{n=1}^{N}\log(\hat{p}_{n,c_n})$ 를 사용한다.

  따라서 드롭아웃과 L2 regularization을 적용한 NN의 loss function은 다음과 같다.

$$\mathcal{L}_{\text{dropout}} := -\frac{1}{N}\sum_{n=1}^{N}\log(\hat{p}_{n,c_n}) + \lambda_1||W_1||_2^2 + \lambda_2||W_2||_2^2 + \lambda_3||b||_2^2$$

- $L_{dropout}$은 앞서 구한 L_{GP-MC}에 마이너스를 취한 것과 같다. 따라서 $L_{dropout}$를 최소화하는 것은 L_{GP-MC}를 최대화하는 것과 같음을 알 수 있다.

$$\mathcal{L}_{\text{GP-MC}} \propto \frac{1}{N}\sum_{n=1}^{N}\hat{p}_{n,c_n} - \frac{p_1}{2N}||M_1||2^2 - \frac{p_2}{2N}||M_2||2^2 - \frac{1}{2N}||m||2^2$$

  즉, 회귀 과제에서와 마찬가지로 분류 과제에서도 NN에 드롭아웃과 L2 regularization을 적용하는 것이 Deep Gaussian Process의 Bayesian Approximation과 같다.

Prior Length-Scale

- NN의 첫 번째 레이어의 가중치 $\boldsymbol{W}_1$에 대해 more informative한 prior를 가정함으로써 데이터의 frequency에 대한 prior belief를 반영할 수 있다.

$$\mathrm{p}(\mathbf{w}) = \mathcal{N}(\boldsymbol{w};\, 0, I_K) \qquad \rightarrow \qquad p_l(\boldsymbol{w}) = \mathcal{N}\left(\mathbf{w}; 0, l^{-2} I_K\right)$$

- Proposition 1을 이용해 variational dist'n $q(\boldsymbol{w})$과 scaling된 prior $p_l(\boldsymbol{w})$ 간의 KL-divergence를 구하면 다음과 같다.

$$\mathrm{KL}(q(x)||p_l(x)) \approx \sum_{i=1}^{L} \frac{p_i}{2}(l^2 \mu_i^T \mu_i + \mathrm{tr}(l^2 \Sigma_i) - K - \log|\Sigma_i| + K \log l^{-2}) + K$$

(참고) Proposition 1. Gaussian Mixture와 Single Gaussian의 KL-분산 근사식

> $Let\ K, L \in \mathbb{N}\ and\ \pi_i$ is probability vector(mixing coefficient)
>
> $$q(\mathbf{x}) = \sum_{i=1}^{L} \pi_i \mathcal{N}(\mathbf{x};\ \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \quad where\ \boldsymbol{\mu}_i \in \mathbb{R}^K,\ \boldsymbol{\Sigma}_i \in \mathbb{R}^{K \times K}$$
>
> $$p(\mathbf{x}) = \mathcal{N}(\mathbf{0}, \mathbf{I}_k)$$
>
> 이때의 KL분산은 다음과 같이 근사 가능하다.
>
> $$\mathrm{KL}(q(\mathbf{x})||p(\mathbf{x})) \approx \sum_{i=1}^{L} \frac{\pi_i}{2}(\mathbf{m}_i^T \mathbf{m}_i + tr(\boldsymbol{\Sigma}_i) - K(1 + \log 2\pi) - \log|\boldsymbol{\Sigma}_i|) + constant$$

- Bias에 대한 prior에도 length-scale을 적용한다: $p_{l'}(\boldsymbol{b}) = \mathcal{N}\left(\mathbf{b}; 0, l'^{-2} I_K\right)$

  앞에서와 같은 방식으로 회귀 과제에서의 maximization objective function을 유도하면 다음과 같다.

$$\mathcal{L}_{\text{GP-MC}} \propto -\frac{1}{2N} \sum_{n=1}^{N} ||\mathbf{y_n} - \hat{\mathbf{y}}_n||_2^2 - \frac{l^2 p_1}{2\tau N} ||\mathbf{M_1}||_2^2 - \frac{p_2}{2\tau N} ||\mathbf{M_2}||_2^2 - \frac{l'^2}{2\tau N} ||\mathbf{m}||_2^2$$

Prior Length-Scale

Length-scaled prior	$\mathcal{L}_{\text{GP-MC}} \propto -\frac{1}{2N}\sum_{n=1}^{N}\|\|\mathbf{y_n}-\hat{\mathbf{y}}_n\|\|_2^2 - \frac{l^2 p_1}{2\tau N}\|\|\mathbf{M_1}\|\|_2^2 - \frac{p_2}{2\tau N}\|\|\mathbf{M_2}\|\|_2^2 - \frac{l'^2}{2\tau N}\|\|\mathbf{m}\|\|_2^2 \qquad \lambda_1 = \frac{l^2 p_1}{2N\tau} \qquad \tau = \frac{l^2 p_1}{2N\lambda_1}$
Unscaled prior	$\mathcal{L}_{\text{GP-MC}} \propto -\frac{1}{2N}\sum_{n=1}^{N}\|\|\mathbf{y_n}-\hat{\mathbf{y}}_n\|\|_2^2 - \frac{p_1}{2\tau N}\|\|\mathbf{M_1}\|\|_2^2 - \frac{p_2}{2\tau N}\|\|\mathbf{M_2}\|\|_2^2 - \frac{1}{2\tau N}\|\|\mathbf{m}\|\|_2^2 \qquad \lambda_1 = \frac{p_1}{2N\tau} \qquad \tau = \frac{p_1}{2N\lambda_1}$

- Length-scale l 을 도입함으로써 Precision parameter τ를 weight-decay λ에서 분리할 수 있다.
  Unscaled prior을 사용한 경우는 τ 가 변하는 비율에 따라 λ 도 함께 변하게 되는데, length-scale을 적용함으로써 그 영향을 분리한다.
- High frequency data → high τ, small l, l' → small λ_1 → weaker regularization over $\boldsymbol{M_1}, \boldsymbol{m}$: 모델이 주어진 데이터에 well-fitted 되게 한다 = high frequency data를 capture 할 수 있다.
  Low frequency data → low τ, large l, l' → large λ_1 → stronger regularization over $\boldsymbol{M_1}, \boldsymbol{m}$: 모델이 데이터에 과적합되지 않도록 한다.

Methodology Summary

Deep GP를 살펴보면 covariance function은 다음과 같다.
(w,b도 이제 분포를 따른다고 가정)

$$\mathbf{K}(\mathrm{x},\mathrm{y}) = \int p(\mathrm{w})p(b)\sigma(\mathbf{w}^T\mathrm{x}+b)\sigma(\mathbf{w}^T\mathrm{y}+b)\mathrm{d}\mathbf{w}\mathrm{d}b$$

Let W_ia random matrix of dimension $K_i \times K_i - 1$

$$\omega = \{\mathbf{W}_i\}_{i=1}^{L}.$$

Deep GP with L layers and covariance function
$K(x,y)$can be approximated by variational inference.

$$p(\mathrm{y}|\mathrm{x},\mathbf{X},\mathbf{Y}) = \int p(\mathrm{y}|\mathrm{x},\omega)p(\omega|\mathbf{X},\mathbf{Y})\mathrm{d}\omega \qquad (2)$$

$$p(\mathrm{y}|\mathrm{x},\omega) = \mathcal{N}(\mathrm{y};\widehat{\mathrm{y}}(\mathrm{x},\omega),\tau^{-1}\mathbf{I}_D)$$

$$\widehat{\mathrm{y}}(\mathrm{x},\omega=\{\mathbf{W}_1,...,\mathbf{W}_L\})$$

$$= \sqrt{\frac{1}{K_L}}\mathbf{W}_L\sigma\left(...\sqrt{\frac{1}{K_1}}\mathbf{W}_2\sigma(\mathbf{W}_1\mathrm{x}+\mathrm{m}_1)...\right)$$

Posterior of deep GP: $p(w|X,Y)$를 구하는 것이 불가능

➔ Variational distribution $q(w)$이용

$$\mathbf{W}_i = \mathbf{M}_i \cdot \mathrm{diag}([\mathrm{z}_{i,j}]_{j=1}^{K_i})$$

$$\mathrm{z}_{i,j} \sim \mathrm{Bernoulli}(p_i) \text{ for } i = 1,...,L,\ j = 1,...,K_{i-1}$$

M_i가 variational parameter가 된다.

$Z_{i,j} = 0$ -> unit j in layer i-1 is dropped out as an input to layer I

Minimize KL divergence between $q(w)$ & $p(w|X,Y)$

$$-\int q(\omega)\log p(\mathbf{Y}|\mathbf{X},\omega)\mathrm{d}\omega + \mathrm{KL}(q(\omega)||p(\omega)).$$

Methodology Summary

Minimize KL divergence between $\boldsymbol{q}(\boldsymbol{w})$ & $\boldsymbol{p}(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{Y})$

$$-\int q(\omega)\log p(\mathbf{Y}|\mathbf{X},\omega)\mathrm{d}\omega + \mathrm{KL}(q(\omega)||p(\omega)).$$

1. First term : Monte Carlo with single sample $w_n \sim q(w)$
2. Scale with $\frac{1}{\tau N}$

$$\mathcal{L}_{\text{GP-MC}} \propto \frac{1}{N}\sum_{n=1}^{N}\frac{-\log p(\mathbf{y}_n|\mathbf{x}_n,\widehat{\omega}_n)}{\tau} + \sum_{i=1}^{L}\left(\frac{p_i l^2}{2\tau N}||\mathbf{M}_i||_2^2 + \frac{l^2}{2\tau N}||\mathbf{m}_i||_2^2\right).$$

NN with L2 Regularization을 사용했을 때의 Loss:

$$\mathcal{L}_{\text{dropout}} := \frac{1}{N}\sum^{N} E(\mathbf{y}_i,\widehat{\mathbf{y}}_i) + \lambda\sum^{L}\left(||\mathbf{W}_i||_2^2 + ||\mathbf{b}_i||_2^2\right)$$

$E(\mathbf{y}_n,\widehat{\mathbf{y}}(\mathbf{x}_n,\widehat{\omega}_n)) = -\log p(\mathbf{y}_n|\mathbf{x}_n,\widehat{\omega}_n)/\tau$ 로 두면 위의 두 Loss가 같다!

즉, NN with dropout & L2 Regularization을 적용하는 것은 Deep Gaussian Process의 Bayesian Approximation으로 해석 가능

4

Mini-Batch Optimization & Deeper Layers

Mini-batch Optimization

1개의 Hidden Layer를 갖는 Neural Networks를 가정하자

그러면 앞부분에서 Evidence Lower bound가 다음과 같이 되는 것을 학습하였다.

$$\mathcal{L}_{\text{GP-MC}} \propto -\frac{1}{2N}\sum_{n=1}^{N}||\mathbf{y}_n - \widehat{\mathbf{y}}_n||_2^2 - \frac{l^2 p_1}{2\tau N}||\mathbf{M}_1||_2^2 - \frac{K p_2}{2\tau N}||\mathbf{M}_2||_2^2 - \frac{l'^2}{2\tau N}||\mathbf{m}||_2^2.$$

Represents likelihood — Represents KL divergence

(어려워 보이지만 사실은 likelihood에서
variational inference를 위한 KL divergence를 뺀 의미에 지나지 않는다.
이 때의 KL divergence는 데이터의 분포 P와 그를 근사할 수 있는 모델 q 사이의 KL divergence이다.)

Mini-batch Optimization

이때, 한 번의 추론에 전체 데이터 샘플 N개를 다 이용하지 않고,

내가 임의로 설정한 집합(Batch)의 원소의 개수 M개를 이용하는 것

이 Minibatch optimization이다.

$$\mathcal{L}_{\text{GP-MC}} \approx -\frac{1}{2M}\sum_{n\in S}||\mathbf{y}_n - \widehat{\mathbf{y}}_n||_2^2 - \frac{p_1}{2\tau N}||\mathbf{M}_1||_2^2 - \frac{p_2}{2\tau N}||\mathbf{M}_2||_2^2 - \frac{1}{2\tau N}||\mathbf{m}||_2^2$$

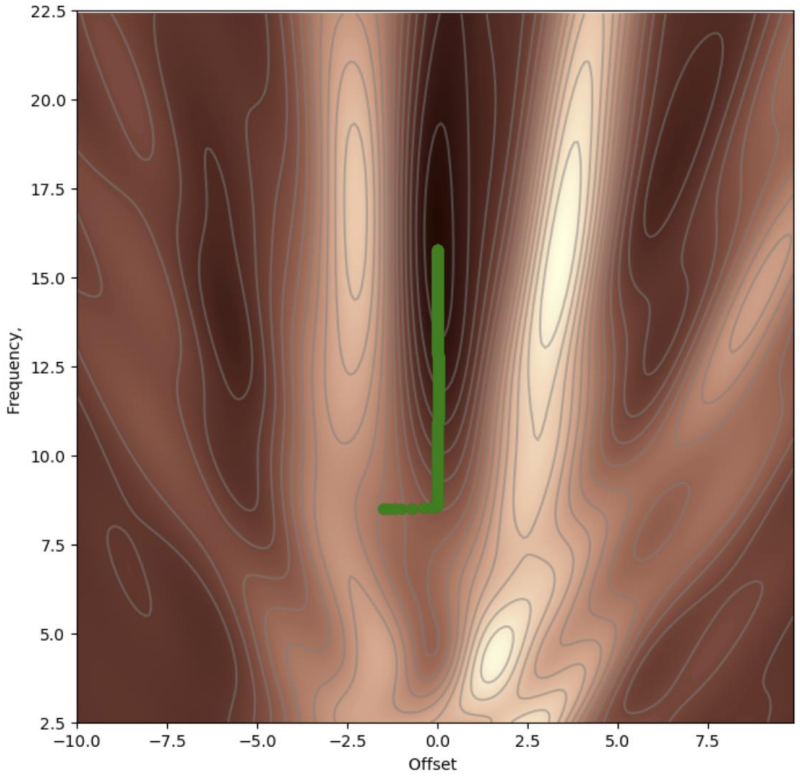
그런데, 실제 분포들 사이의 차이를 나타내는

KL Divergence는 그대로 N을 이용하듯이,

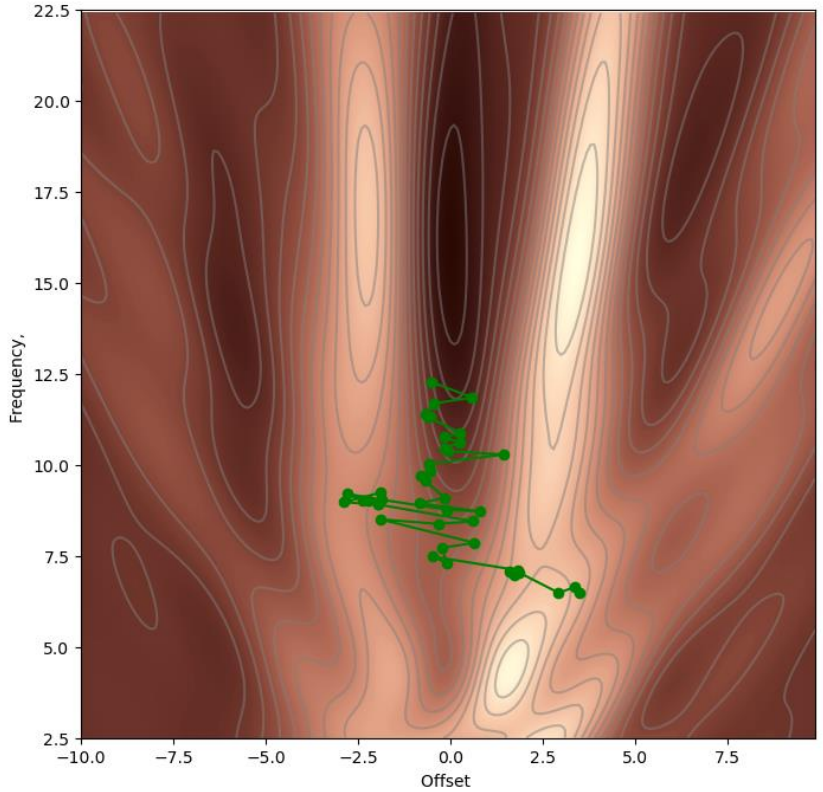
M개의 데이터는 전체 N개의 데이터세트를 완전하게 근사하지 못하기 때문에

트레이닝 에러는 일반적으로 증가하게 된다.

Mini-batch Optimization



Gradient Descent



Mini-batch Gradient Descent

그림에서 볼 수 있듯이, Minibatch를 이용하면 전체 데이터세트의 분포와 달라지기 때문에,
최적화가 불안정하게 일어나는 것을 확인할 수 있다.

Mini-batch Optimization

그렇지만 딥러닝을 조금이라도 공부해 본 사람은,

최적화 시 한 시행에서 전체 데이터를 다 이용하는 것이 Batch GD 대신,

반드시 부분 데이터를 이용하는 Minibatch GD(Gradient Descent)를

최적화 방법으로 선택한다는 것을 배웠을 것이다.

이는, Minibatch GD가 많은 데이터를 한 번에 이용하는 GD에 비해
적은 데이터로 이루어진 여러 집합을 이용하기 때문에

앙상블(Ensemble)효과가 발생해
Generalization Error(Test Error)는 오히려 감소하기 때문이다.

즉, Minibatch를 이용하면, 모델의 성능을 향상시키는 데 큰 도움이 된다.

Mini-batch Optimization

이처럼, 신경망 학습에는 Test Error를 Training Error보다 적게 만드는
Generalization을 잘 하는 것이 무엇보다 중요한데,

이는 Minibatch와 같이 복잡한 전체를
간단한 부분들의 합으로 분해하는 것으로 향상시킬 수 있다.
간단한 부분들을 앙상블하는 것이 복잡한 전체보다 더 높은 일반화 능력을 보이기 때문이다.

Minibatch가 아니더라도
이번 발표의 핵심 주제인 Dropout부터,
또 널리 사용되는 Regularization 방법인 L1, L2 Decay 모두 이러한 원리를 이용한다.

Predictive Log-likelihood

분류 문제를 위한 신경망 학습에서는 다음과 같은 Likelihood 분포를 최대화해야 한다.

$$\log p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y})$$

이는, 내가 가진 Dataset X와 Y에서,

새로운 데이터 point x*이 입력되었을 때,

출력을 올바르게 y*로 분류할 확률을 의미한다.

Predictive Log-likelihood

그런데 논문을 읽어 보면, log likelihood를

다음과 같은 괴상한 식으로 분해하는 것을 확인할 수 있다.

$$\begin{aligned}\log p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) &= \log \int p(\mathbf{y}^*|\mathbf{x}^*, \boldsymbol{\omega}) p(\boldsymbol{\omega}|\mathbf{X}, \mathbf{Y}) \mathrm{d}\boldsymbol{\omega} \\ &\approx \log \int p(\mathbf{y}^*|\mathbf{x}^*, \boldsymbol{\omega}) q(\boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega} \\ &\approx \log \left(\frac{1}{T} \sum_{t=1}^{T} p(y^*|\mathbf{x}^*, \boldsymbol{\omega}_t) \right)\end{aligned}$$

이번 Appendix의 목표는, 위 식을 이해하는 것이다.

Predictive Log-likelihood

앞의 식을 이해하기 위해 알면 좋은 사전 지식은

Coding에 대해서 이해하는 것이다.

Coding이라고 하면 Python과 같은 프로그래밍 언어를 먼저 떠올리는 것이 일반적이지만

Coding은 본래 일종의 Transformation(변환)을 의미한다.

이때, 정보를(Complexity)를 압축하는 변환을 인코딩(Encoding),

정보를 늘리는 변환을 디코딩(Decoding)이라 한다.

Predictive Log-likelihood

여기서는 가중치(Weight) W의 역할에 대해 초점을 맞춘다.

가중치 W는 데이터 분포 P(X)를 압축하는 역할을 하는데,

우리는 엄청나게 복잡한 데이터의 확률분포를

비교적 적은 수의 변수인 W로 표현할 수 있다.

즉, 가중치 W는 데이터 (X,Y)의 Encoding이라고 볼 수 있으며
이러한 Encoding 과정은, 표현하기 힘든 변수를 근사적으로 표현하는 데 도움을 준다.

Deeper Networks

이제 코딩 이론의 관점에서 다음 식을 다시 살펴보자.

$$\log p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) = \log \int p(\mathbf{y}^*|\mathbf{x}^*, \boldsymbol{\omega}) p(\boldsymbol{\omega}|\mathbf{X}, \mathbf{Y}) \mathrm{d}\boldsymbol{\omega}$$

$$\approx \log \int p(\mathbf{y}^*|\mathbf{x}^*, \boldsymbol{\omega}) q(\boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega}$$

$$\approx \log \left(\frac{1}{T} \sum_{t=1}^{T} p(y^*|\mathbf{x}^*, \boldsymbol{\omega}_t) \right)$$

내가 가진 Dataset로
새로운 Data를
잘 예측할 수 있는 능력

2. 인코딩 된 w(우리의 Model)를
바탕으로 새로운 데이터 예측 및 분류

1. 데이터 X,Y를 먼저 가중치 w로 인코딩

3. 그러나, 좋은 w를 한번에 구하는 것은
Computational Cost (O(N^3)) 가 많이 소비되기 때문에,
Variational Inference를 통해
P의 근사인 q를 통해 간접적으로 p를 예측

4. Monte Carlo 근사를 통해, Integration을 Summation으로 바꿈
이는 우리가 일반적으로 이용하는 Loss Function의 형태와 유사하며,
좋은 Model을 얻기 위해서는 데이터를 잘 모사하는 Encoding(Weight)을
해야 한다는 것을 시사함

감사합니다