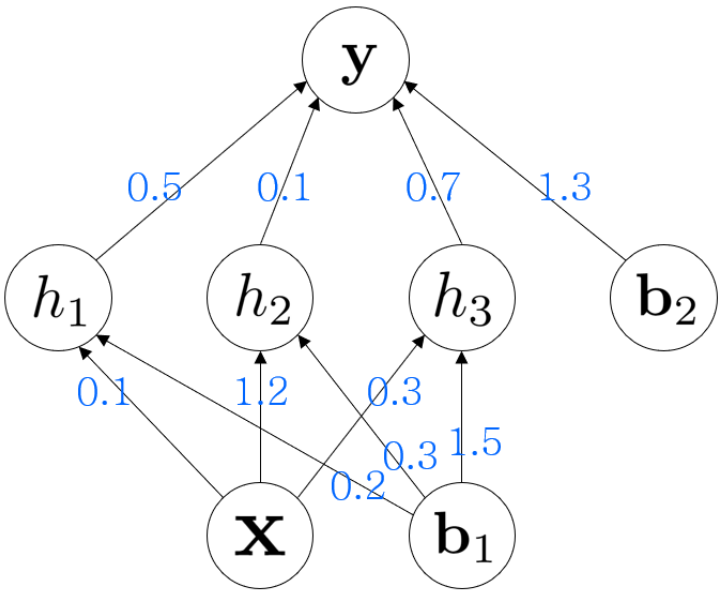
Dropout as a Bayesian Approximation:
Representing Model Uncertainty in Deep Learning

정몽주, 박태준

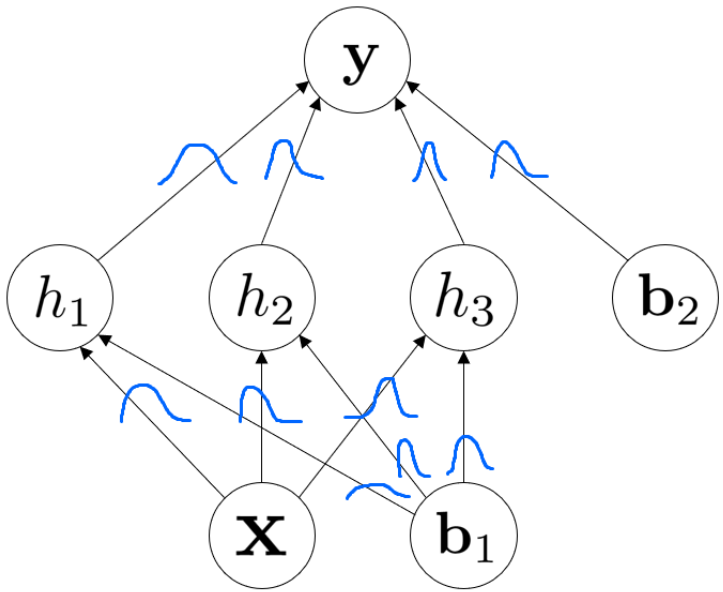
공간통계연구실

2022년 8월

Bayesian Neural Networks



Standard Neural Network



Bayesian Neural Network

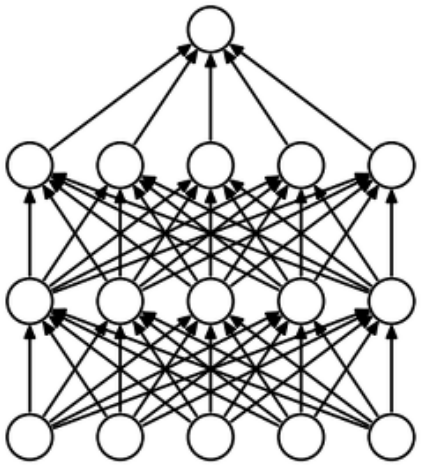
Bayesian Neural Networks

- **Bayesian neural networks (BNNs, Bayesian NNs)** place a prior distribution over a neural network's weights.

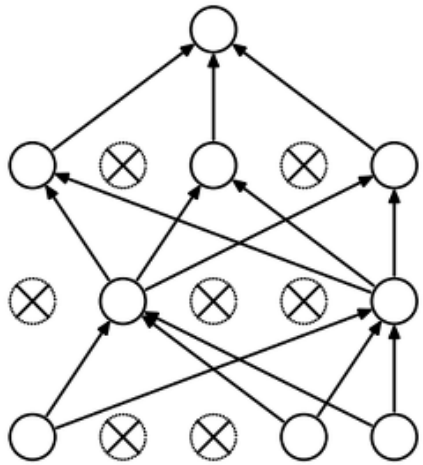
$$p(\boldsymbol{\omega})$$

- Bayesian neural networks offer a probabilistic interpretation of deep learning model.
  - **Model uncertainty**

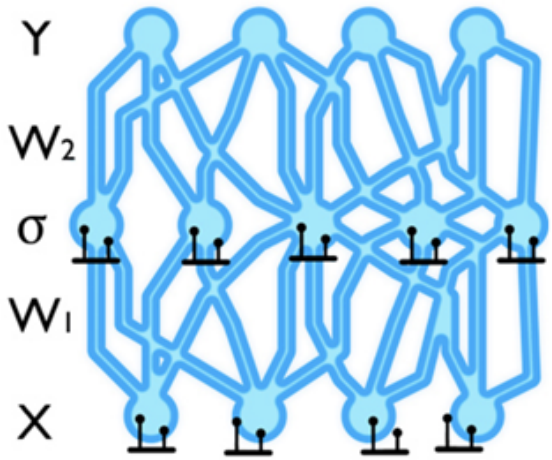
Dropout



(a) Standard Neural Net



(b) After applying dropout.



- **Dropout** is used in many models in deep learning as a way to avoid over-fitting.
- The key idea is to randomly drop units (along with their connections) from the neural network during training.

Topic

Dropout as a Bayesian Approximation

- A neural network with dropout applied before every weight layer is mathematically equivalent to an approximation to a well known Bayesian model : the Gaussian process (GP).
- We develop tools for representing model uncertainty of existing dropout NNs.

Contents

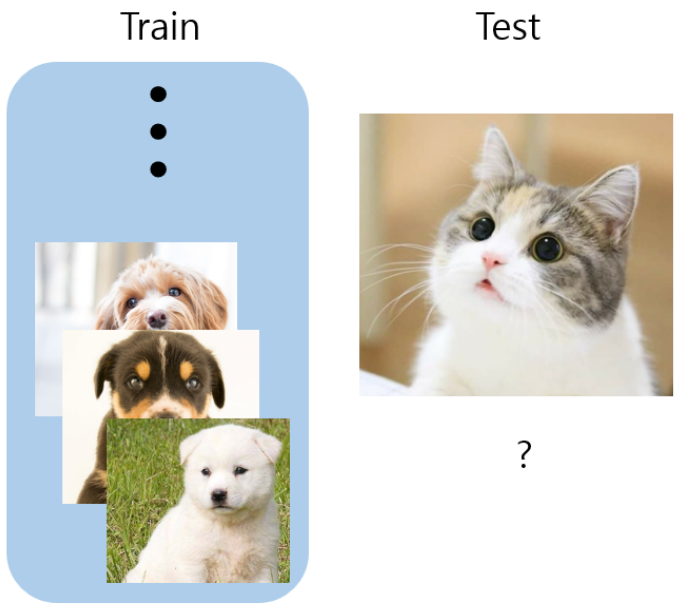
Section 1

Introduction

1. Introduction

Model Uncertainty

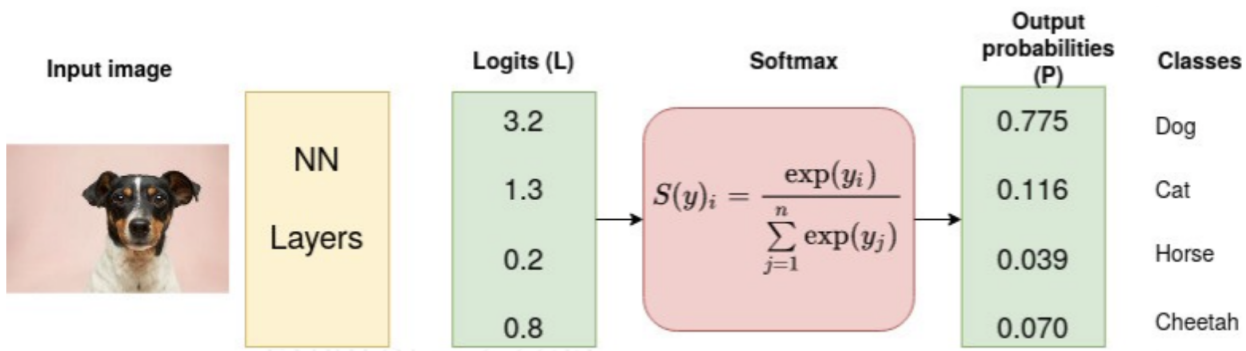
- **Out of distribution test data:**
  - Given several pictures of dog breeds as training data
  - What should happen if a user uploads a photo of a cat and asks the website to decide on a dog breed?



1. Introduction

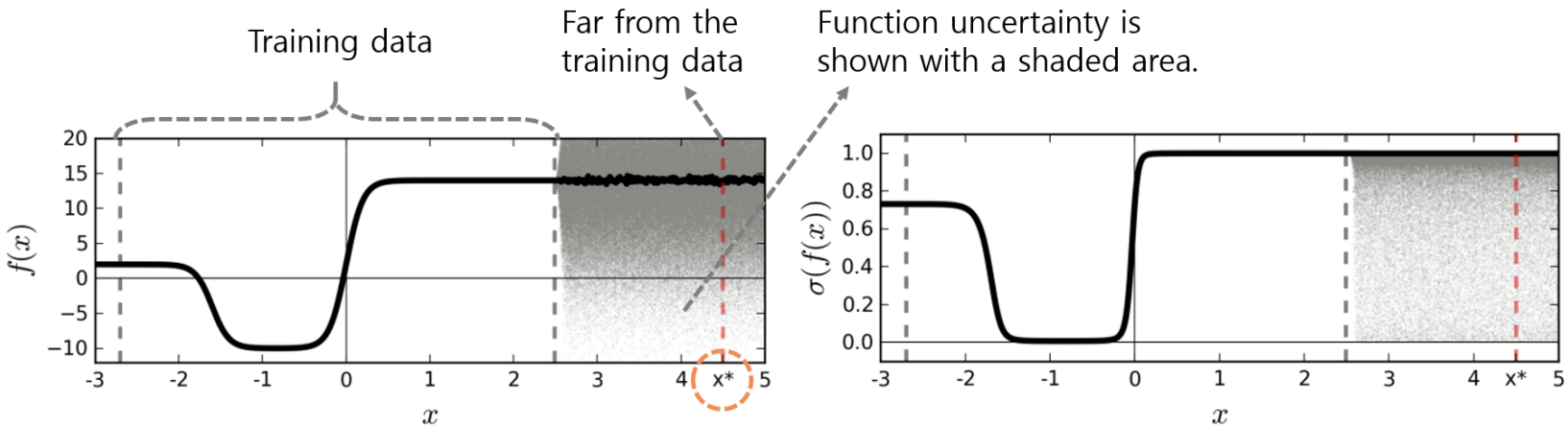
Model Uncertainty

- Softmax function converts a vector of K real numbers into a probability distribution of K possible outcomes.
- The softmax function is often used as the last activation function of a neural network to normalize the output of a network to a probability distribution over predicted output classes.



1. Introduction

Model Uncertainty



(a) Arbitrary function $f(\mathbf{x})$ as a function of data $\mathbf{x}$ (softmax *input*)

(b) $\sigma(f(\mathbf{x}))$ as a function of data $\mathbf{x}$ (softmax *output*)

Figure: **A sketch of softmax input and output for an idealised binary classification problem.** Ignoring function uncertainty, point x^* is classified as class 1 with probability 1.

- Standard deep learning tools for regression and classification do not capture model uncertainty.

1. Introduction

Model Uncertainty

- Representing model uncertainty is of crucial importance.
- With model confidence at hand we can treat uncertain inputs and special cases explicitly.
  - For example, in the case of classification, a model might return a result with high uncertainty.
  - In this case, we might decide to pass the input to a human for classification.

Section 2

Background

2. Background

Notation

- $\mathbf{x}$: Vectors.
- $\mathbf{X}$: Matrices.
- x: scalar quantities.
- $\mathbf{x}_i$: Entire rows or columns.
- x_{ij}: Specific elements.

- $\mathbf{W}_1 : Q \times K,\ \mathbf{W}_2 : K \times D$: Variables.
- $\mathbf{w}_q,\ \mathbf{w}_k$: Corresponding lower case indices to refer to specific rows / columns for the first variable.
- $\mathbf{w}_k,\ \mathbf{w}_d$: For the second variable.

- $w_{1,qk}$: The element at row q column k of the variable $\mathbf{W}_1$.
- $w_{2,kd}$: The element at row k column d of the variable $\mathbf{W}_2$.

Subsection 1

Dropout

2. Background

2.1 Dropout

- Review the dropout NN model for the case of a single hidden layer NN.
- The generalization to multiple layers is straightforward.

- $\mathbf{W}_1$ $(Q \times K)$: The weight matrix connecting the first layer to the hidden layer.
- $\mathbf{W}_2$ $(K \times D)$: The weight matrix connecting connecting the hidden layer to the output layer.
- $\mathbf{b}$: The biases, K dimensional vector.
- $\sigma(\cdot)$: Activation function, some element-wise non-linearity.

- **A standard NN model would output**:

$$\hat{\mathbf{y}} = \sigma(\mathbf{x}\mathbf{W}_1 + \mathbf{b})\mathbf{W}_2$$

given some input $\mathbf{x}$.

  - Note that we omit the outer-most bias term as this is equivalent to centring the output.

2. Background

2.1 Dropout

- Dropout is applied by sampling two binary vectors $\mathbf{z}_1, \mathbf{z}_2$ of dimensions Q and K respectively, with

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

- Given an input $\mathbf{x}$, $1 - p_1$ proportion of the elements of the input are set to zero:

$$\mathbf{x} \circ \mathbf{z}_1.$$

- **The dropout model's output**:

$$\hat{\mathbf{y}} = \sigma\Big(\mathbf{x}(\mathbf{z}_1\mathbf{W}_1) + \mathbf{b}\Big)(\mathbf{z}_2\mathbf{W}_2).$$

  - (We will write $\mathbf{z}_1$ when we mean $\text{diag}(\mathbf{z}_1)$.)

2. Background

2.1 Dropout

- $\{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$: N observed inputs.
- $\{\mathbf{y}_1, \cdots, \mathbf{y}_N\}$: Corresponding observed outputs.
- $\{\hat{\mathbf{y}}_1, \cdots, \hat{\mathbf{y}}_N\}$: The outputs of the model.

- We often use L_2 regularization for parameters $\{\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}\}$ weighted by some weight decays λ_i.

- **Minimization objective (cost)** for regression:

$$\mathcal{L}_{\text{dropout}} := \frac{1}{2N} \sum_{n=1}^{N} ||\mathbf{y}_n - \hat{\mathbf{y}}_n||_2^2 + \lambda_1 ||\mathbf{W}_1||_2^2 + \lambda_2 ||\mathbf{W}_2||_2^2 + \lambda_3 ||\mathbf{W}_3||_2^2$$

Subsection 2

Gaussian Processes

- We use a **Gaussian process (GP)** to describe a distribution over functions.
- The Gaussian process offers desirable properties such as uncertainty estimates over the function values.

Definition (Gaussian Process)

A random process $X(t)$ is a **Gaussian process** if $^{\forall}k \in \mathbb{N}$, $^{\forall}t_1, \cdots, t_k \in \mathcal{T}$, a random vector formed by $X(t_1), \cdots, X(t_k)$ is jointly Gaussian.

- A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.

- The joint density is completely specified by
  - **Mean function**: $m(t) = \mathbb{E}[X(t)]$, where $m(\cdot)$ is known as a mean function.
  - **Covariance function**: $k(t, s) = \mathrm{Cov}(X(t), X(s))$, where $k(\cdot, \cdot)$ is known as a covariance function.
- **Notation**: $X(t) \sim \mathcal{GP}(m(t), k(t, s))$

- $\{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$, $\{\mathbf{y}_1, \cdots, \mathbf{y}_N\}$: Training dataset.
- $\mathbf{X} \in \mathbb{R}^{N \times Q}$, $\mathbf{Y} \in \mathbb{R}^{N \times D}$: Inputs and outputs.

- We would like to estimate a function

$$\mathbf{y} = \mathbf{f}(\mathbf{x})$$

  that is likely to have generated our observation.
- Following the Bayesian approach, we would put some **prior distribution** over the space of functions $p(\mathbf{f})$.

- We then look for the **posterior distribution** over the space of functions given our dataset

$$p(\mathbf{f}|\mathbf{X}, \mathbf{Y}) \propto p(\mathbf{Y}|\mathbf{X}, \mathbf{f})p(\mathbf{f}).$$

- In our case the random variables represent the value of the function $\mathbf{f}(\mathbf{x})$ at location $\mathbf{x}$.

Remark (Describe a distribution over functions)

- A Gaussian process is completely specified by its mean function and covariance function.
- Define mean function and the covariance function of $\mathbf{f}(\mathbf{x})$ as

$$m(\mathbf{x}) = \mathbb{E}[\mathbf{f}(\mathbf{x})]$$
$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(\mathbf{f}(\mathbf{x}) - m(\mathbf{x}))(\mathbf{f}(\mathbf{x}') - m(\mathbf{x}'))^\top]$$

- Write the Gaussian process as

$$\mathbf{f}(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')).$$

- In practice, we place a joint Gaussian distribution over all function values

$$\mathbf{F}|\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}(\mathbf{X}, \mathbf{X}))$$

- To model the data we have to choose a covariance function $\mathbf{K}(\mathbf{X}_1, \mathbf{X}_2)$ for the Gaussian distribution.

Subsection 3

Variational Inference

- we could condition the model on a finite set of random variables $\boldsymbol{\omega}$.
- Goal of a Bayesian neural network is to find a posterior distribution:

$$p(\boldsymbol{\omega}|\mathbf{X}, \mathbf{Y}) \propto p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\omega})p(\boldsymbol{\omega})$$

where $\mathbf{X}$ and $\mathbf{Y}$ are the input and output training data and $\boldsymbol{\omega}$ is a set of parameters of our interest.

- Once we have $p(\boldsymbol{\omega}|\mathbf{X}, \mathbf{Y})$, the output $\mathbf{y}^* \in \mathbb{R}^D$ at unseen $\mathbf{x}^* \in \mathbb{R}^Q$ is predicted as:

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

  called **the posterior predictive distribution** for a new $\mathbf{x}^*$.

- In practice, none of them is tractable.☹

- **Variational inference** is often used to handle this issue.☺

- The distribution $p(\boldsymbol{\omega}|\mathbf{X}, \mathbf{Y})$ cannot usually be evaluated analytically.
- Instead we define an approximating variational distribution $q(\boldsymbol{\omega})$, whose structure is easy to evaluate.

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

$$\Downarrow$$

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

- Instead of finding $p(\boldsymbol{\omega}|\mathbf{X}, \mathbf{Y})$, optimize $q(\boldsymbol{\omega})$ by minimizing the Kullback–Leibler divergence

$$\mathrm{KL}\Big(q(\boldsymbol{\omega})\Big\|p(\boldsymbol{\omega}|\mathbf{X}, \mathbf{Y})\Big) = \int q(\boldsymbol{\omega}) \log \frac{q(\boldsymbol{\omega})}{p(\boldsymbol{\omega}|\mathbf{X}, \mathbf{Y})} d\boldsymbol{\omega}.$$

- $\mathrm{KL}(q(\boldsymbol{\omega})\|p(\boldsymbol{\omega}|\mathbf{X}, \mathbf{Y}))$ is not tractable as well.☹

- However, minimizing the Kullback–Leibler divergence is equivalent to maximizing the **log evidence lower bound (ELBO)**,☺

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

$$\Big(\leq \log p(\mathbf{Y}|\mathbf{X}) = \text{'log evidence'}\Big),$$

with respect to the variational parameters defining $q(\boldsymbol{\omega})$.

2. Background

2.3 Variational Inference

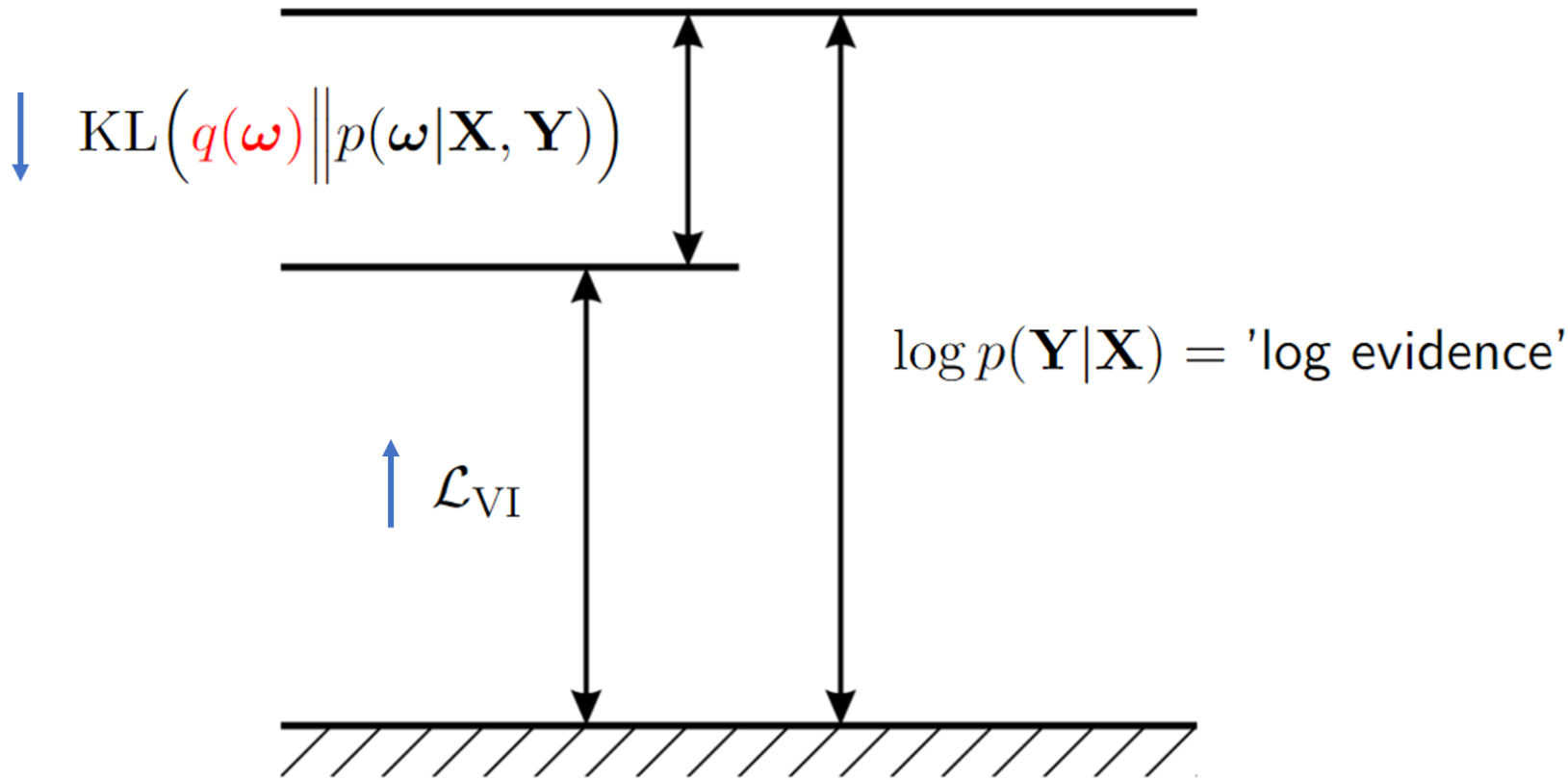


Figure: the ELBO, $\mathcal{L}_{\mathrm{VI}}$, for minimizing KL divergence.

- So in computing ELBO, we don't need to know $p(\boldsymbol{\omega}|\mathbf{X}, \mathbf{Y})$.☺

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

- All we need is
  - Prior: $p(\boldsymbol{\omega})$
  - Variational distribution: $q(\boldsymbol{\omega})$
  - Likelihood: $p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\omega})$
    - For defining likelihood function, we use a Gaussian process.
    - So we need a covariance function $\mathbf{K}(\cdot, \cdot)$.

Section 3

Dropout as a Bayesian Approximation

3. Dropout as a Bayesian Approximation

- We will show that deep NNs with dropout applied before every weight layer are mathematically equivalent to approximate variational inference in the deep Gaussian process.

- Starting with the full Gaussian process we will develop an approximation optimization objective, $\mathcal{L}_{\mathrm{GP-MC}}$, and show that

$$\frac{\partial}{\partial \theta}\mathcal{L}_{\mathrm{dropout}}(\theta) = C \cdot \frac{\partial}{\partial \theta}\mathcal{L}_{\mathrm{GP-MC}}(\theta) \quad \text{(Equiv. in function)}$$

  for the parameters θ , some constant C.

Subsection 1

A Gaussian Process Approximation

3. Dropout as a Bayesian Approximation

3.1 A Gaussian Process Approximation

- We begin by defining our covariance function:

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

  where $p(\mathbf{w})$ is a standard multivariate normal distribution of dimensionality Q and $p(b)$ is a some distribution.

- This defines a valid covariance function. (Tsuda et al., 2002)

3. Dropout as a Bayesian Approximation

3.1 A Gaussian Process Approximation

- We use Monte Carlo integration with K terms:

$$\hat{\mathbf{K}}(\mathbf{x}, \mathbf{y}) = \frac{1}{K} \sum_{k=1}^{K} \sigma(\mathbf{w}_k^\top \mathbf{x} + b_k) \sigma(\mathbf{w}_k^\top \mathbf{y} + b_k)$$

  with $\mathbf{w}_k \sim p(\mathbf{w}), b_k \sim p(b)$.

- K will be the number of hidden units in our single hidden layer NN approximation.

3. Dropout as a Bayesian Approximation

3.1 A Gaussian Process Approximation

- Using $\hat{\mathbf{K}}$ instead of $\mathbf{K}$ as the covariance function of the Gaussian process yields:

$$\begin{aligned}
\mathbf{w}_k &\sim p(\mathbf{w}), b_k \sim p(b), \\
\mathbf{W}_1 &= [\mathbf{w}_k]_{k=1}^K, \mathbf{b} = [b_k]_{k=1}^K \\
\hat{\mathbf{K}}(\mathbf{x}, \mathbf{y}) &= \frac{1}{K} \sum_{k=1}^K \sigma(\mathbf{w}_k^\top \mathbf{x} + b_k) \sigma(\mathbf{w}_k^\top \mathbf{y} + b_k) \\
\mathbf{F} | \mathbf{X}, \mathbf{W}_1, \mathbf{b} &\sim \mathcal{N}(\mathbf{0}, \hat{\mathbf{K}}(\mathbf{X}, \mathbf{X})) \\
\mathbf{y} | \mathbf{F} &\sim \mathcal{N}(\mathbf{F}, \tau^{-1} \mathbf{I}_N),
\end{aligned}$$

with $\mathbf{W}_1$ a $Q \times K$ matrix parametrizing our covariance function.

3. Dropout as a Bayesian Approximation

3.1 A Gaussian Process Approximation

- Suppose $\phi(\mathbf{x}) \in \mathbb{R}^{1\times K}$ is a hidden layer row vector coreesponding to $\mathbf{x} \in \mathbb{R}^{Q}$:

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

- Then

$$\Phi = [\phi(\mathbf{x}_n, \mathbf{W}_1, \mathbf{b})]_{n=1}^{N}$$

  is a $N \times K$ feature matrix.

- We have

$$\hat{\mathbf{K}}(\mathbf{X}, \mathbf{X}) = \Phi\Phi^\top.$$

- Then

$$\mathbf{y}|\mathbf{W}_1, \mathbf{b}, \mathbf{X} \sim N(\mathbf{0}, \Phi\Phi^\top + \tau^{-1}\mathbf{I}_N) \quad \text{where } \mathbf{y} \in \mathbb{R}^{N\times 1}$$

3. Dropout as a Bayesian Approximation

3.1 A Gaussian Process Approximation

- Assume that the output dimensions of a multi-output GP are independent.
- Note that

$$\mathbf{Y} = \begin{pmatrix} | & | & & | \\ \mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_D \\ | & | & & | \end{pmatrix} \in \mathbb{R}^{N \times D}$$

where $\mathbf{y}_d | \cdots \sim \mathcal{N}(0, \Phi\Phi^\top + \tau^{-1}\mathbf{I}_N),\ d = 1, \cdots, D$

- Introduce a $K \times 1$ auxiliary random variable $\mathbf{w}_d \sim \mathcal{N}(0, \mathbf{I}_K)$. Then $\mathbf{y}_d$ has the following:

$$\begin{aligned} &\mathcal{N}(\mathbf{y}_d; 0, \Phi\Phi^\top + \tau^{-1}\mathbf{I}_N) \\ &\qquad = \int \mathcal{N}(\mathbf{y}_d; \Phi\mathbf{w}_d, \tau^{-1}\mathbf{I}_N)\mathcal{N}(\mathbf{w}_d; 0, \mathbf{I}_K) d\mathbf{w}_d. \end{aligned}$$

3. Dropout as a Bayesian Approximation

3.1 A Gaussian Process Approximation

- Writing $\mathbf{W}_2 = [\mathbf{w}_d]_{d=1}^D \in \mathbb{R}^{K \times D}$ then we finally have the following likelihood:

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

where the integration is w.r.t. $\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}$.

Subsection 2

Log Evidence Lower Bound Optimization

3. Dropout as a Bayesian Approximation

3.2 Log Evidence Lower Bound Optimization

- Define $q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) := q(\mathbf{W}_1)q(\mathbf{W}_2)q(\mathbf{b})$
- Define $q(\mathbf{W}_1)$ and $q(\mathbf{W}_2)$ to be a Gaussian mixture distribution with two components, factorised over Q and K.

$$
\begin{aligned}
q(\mathbf{W}_1) &= \prod_{q=1}^{Q} q(\mathbf{w}_d), \quad q(\mathbf{W}_2) = \prod_{k=1}^{K} q(\mathbf{w}_k) \\
q(\mathbf{w}_q) &= p_1 N(\mathbf{m}_q, \sigma^2 \mathbf{I}_K) + (1 - p_1) N(0, \sigma^2 \mathbf{I}_K) \\
q(\mathbf{w}_k) &= p_2 N(\mathbf{m}_k, \sigma^2 \mathbf{I}_D) + (1 - p_2) N(0, \sigma^2 \mathbf{I}_D)
\end{aligned}
$$

  with some probability $p_1, p_2 \in [0, 1]$, scalar $\sigma > 0$ and $\mathbf{m}_q \in \mathbb{R}^K$.
- $q(\mathbf{b}) = N(\mathbf{m}, \sigma^2 \mathbf{I}_K)$.

3. Dropout as a Bayesian Approximation

3.2 Log Evidence Lower Bound Optimization

- Now, recall ELBO:

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

- To sum up, we have

$$\begin{aligned}
\mathcal{L}_{\text{GP}-\text{VI}} &= \int q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) \log p(\mathbf{Y}|\mathbf{X}, \mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) \\
&\quad - \text{KL}\Big(q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) \Big\| p(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b})\Big) \\
&= \sum_{n=1}^{N} \int q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) \log p(\mathbf{y}_n|\mathbf{x}_n, \mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) \\
&\quad - \text{KL}\Big(q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) \Big\| p(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b})\Big)
\end{aligned}$$

3. Dropout as a Bayesian Approximation

3.2 Log Evidence Lower Bound Optimization

- Rewrite the $\log p(\mathbf{Y}|\mathbf{X}, \mathbf{W}_1, \mathbf{W}_2, \mathbf{b})$ as a sum :

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

, here $\hat{\mathbf{y}}_n = \phi(x_n, \mathbf{W}_1, \mathbf{b})\mathbf{W}_2$.

3. Dropout as a Bayesian Approximation

3.2 Log Evidence Lower Bound Optimization

- We estimate each integral using Monte Carlo integration:

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

with realizations

$$\begin{aligned}
\mathbf{W}_1^n &= \mathbf{z}_1^n(\mathbf{M}_1 + \boldsymbol{\sigma}\boldsymbol{\epsilon}_1^n) + (1 - \mathbf{z}_1^n)\boldsymbol{\sigma}\widehat{\boldsymbol{\epsilon}}_1^n, \\
\mathbf{W}_2^n &= \mathbf{z}_2^n(\mathbf{M}_2 + \boldsymbol{\sigma}\boldsymbol{\epsilon}_2^n) + (1 - \mathbf{z}_2^n)\boldsymbol{\sigma}\widehat{\boldsymbol{\epsilon}}_2^n, \\
\mathbf{b}^n &= \mathbf{m} + \boldsymbol{\sigma}\boldsymbol{\epsilon}^n,
\end{aligned}$$

where

$$\begin{aligned}
\boldsymbol{\epsilon}_1^n &\sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{Q \times K}), \quad & \mathbf{z}_{1,q} &\sim \mathrm{Bernoulli}(p_1) \\
\boldsymbol{\epsilon}_2^n &\sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{K \times D}), \quad & \mathbf{z}_{2,q} &\sim \mathrm{Bernoulli}(p_2)
\end{aligned}$$

3. Dropout as a Bayesian Approximation

3.2 Log Evidence Lower Bound Optimization

- Then, We can re-write $\mathcal{L}_{\mathrm{GP-VI}}$ as

$$\sum_{n=1}^{N} \int q(\mathbf{z}_1, \boldsymbol{\epsilon}_1, \mathbf{z}_2, \boldsymbol{\epsilon}_2, \boldsymbol{\epsilon}) \log p(\mathbf{Y}|\mathbf{X}, \mathbf{W}_1(\mathbf{z}_1, \boldsymbol{\epsilon}_1), \mathbf{W}_2(\mathbf{z}_2, \boldsymbol{\epsilon}_2), \mathbf{b}(\boldsymbol{\epsilon}))$$
$$- \mathrm{KL}\Big(q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) \Big\| p(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b})\Big)$$

  where each integration is over $\boldsymbol{\epsilon}_1, \mathbf{z}_1, \boldsymbol{\epsilon}_2, \mathbf{z}_2, \boldsymbol{\epsilon}$.

- Estimate each integral using MC integration with a distinct single sample to obtain :

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

3. Dropout as a Bayesian Approximation

3.2 Log Evidence Lower Bound Optimization

- Optimising the stochastic objective $\mathcal{L}_{\mathrm{GP-MC}}$ we would converge to the same limit as $\mathcal{L}_{\mathrm{GP-VI}}$.

- We can't evaluate the KL divergence term between a mixture of Gaussians and a single Gaussian analytically.
- However we can perform Monte Carlo integration like in the above.

3. Dropout as a Bayesian Approximation

3.2 Log Evidence Lower Bound Optimization

- In real-world $\boldsymbol{\sigma}$ will be represented on a computer, in effect, as zero, resulting in

$$\widehat{\mathbf{W}}_1^n \approx \hat{\mathbf{z}}_1^n \mathbf{M}_1, \ \widehat{\mathbf{W}}_2^n \approx \hat{\mathbf{z}}_2^n \mathbf{M}_2, \widehat{\mathbf{b}}^n \approx \mathbf{m}.$$

- With Gaussian prior on the parameters $\mathbf{W}1$, $\mathbf{W}2$, and $\mathbf{b}_1$, we obtain the maximization objective:

$$\begin{aligned}
&\mathcal{L}_{\mathrm{GP-MC}} \\
&\propto {}^{1} -\frac{\tau}{2}\sum_{n=1}^{N}||\mathbf{y}_n - \hat{\mathbf{y}}_n||_2^2 - \mathrm{KL}\Big(q(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b})\Big\|p(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b})\Big) \\
&\approx -\frac{\tau}{2}\sum_{n=1}^{N}||\mathbf{y}_n - \hat{\mathbf{y}}_n||_2^2 - \frac{p_1}{2}||\mathbf{M}_1||_2^2 - \frac{p_2}{2}||\mathbf{M}_2||_2^2 - \frac{1}{2}||\mathbf{m}||_2^2
\end{aligned}$$

[1] Abused notation $A \propto B$ to mean $A = B + c$ rather than $A = cB$ for some constant c, in this paper.

3. Dropout as a Bayesian Approximation

3.2 Log Evidence Lower Bound Optimization

- So, Maximising $\mathcal{L}_{\mathrm{GP-MC}}$ results in the same optimal parameters as the minimisation of $\mathcal{L}_{\mathrm{dropout}}$:

$$\frac{\partial}{\partial \theta}\mathcal{L}_{\mathrm{dropout}}(\theta) = -\frac{1}{\tau N}\frac{\partial}{\partial \theta}\mathcal{L}_{\mathrm{GP-MC}}(\theta)$$

  for the parameters θ.

Subsection 3

Going Deeper than a Single Hidden Layer

3. Dropout as a Bayesian Approximation

3.3 How to extend the derivation to two hidden layers?

- Define a different covariance function for the GPs in the 2nd layers :

$$\mathbf{K}_2(\mathbf{x}, \mathbf{y}) = \frac{1}{K_2} \int p(\mathbf{b}_2) \sigma_2(\mathbf{x} + \mathbf{b}_2)^\top \sigma_2(\mathbf{y} + \mathbf{b}_2) d\mathbf{b}_2$$

  with some distribution $p(\mathbf{b}_2)$ over $\mathbf{b}_2 \in \mathbb{R}^{K_2}$.

- We use MC integration with one term :

$$\hat{\mathbf{K}}_2(\mathbf{x}, \mathbf{y}) = \frac{1}{K_2} \sigma_2(\mathbf{x} + \mathbf{b}_2)^\top \sigma_2(\mathbf{y} + \mathbf{b}_2)$$

3. Dropout as a Bayesian Approximation

3.3 How to extend the derivation to two hidden layers?

- We generate the model's output

$$\mathbf{F}_1|\mathbf{X}, \mathbf{W}_1, \mathbf{b}_1 \sim \mathcal{N}(\mathbf{0}, \hat{\mathbf{K}}_1(\mathbf{X}, \mathbf{X}))$$
$$\mathbf{F}_2|\mathbf{X}, \mathbf{b}_2 \sim \mathcal{N}(\mathbf{0}, \hat{\mathbf{K}}_2(\mathbf{F}_1, \mathbf{F}_1))$$
$$\mathbf{Y}|\mathbf{F}_2 \sim \mathcal{N}(\mathbf{F}_2, \tau^{-1}\mathbf{I}_N)$$

- We introduce auxiliary random variables $\mathbf{W}_2 \in \mathbb{R}^{K_1 \times K_2}$, $\mathbf{W}_3 \in \mathbb{R}^{K_2 \times D}$. The columns of each matrix distribute according to $\mathcal{N}(0, \mathbf{I})$

3. Dropout as a Bayesian Approximation

3.3 How to extend the derivation to two hidden layers?

- Like before, write $\hat{\mathbf{K}}_1(\mathbf{X}, \mathbf{X}) = \Phi_1 \Phi_1^\top$ with $\Phi_1 \in \mathbb{R}^{N \times K_1}$, then we can write $\mathbf{F}_1 = \Phi_1 \mathbf{W}_2$.
- Let $\hat{\mathbf{K}}_2(\mathbf{X}, \mathbf{X}) = \Phi_2 \Phi_2^\top$ with $\Phi_2 \in \mathbb{R}^{N \times K_2}$:

$$\phi_{2,nk} = \sqrt{\frac{1}{K_2}} \sigma_2 \left(\mathbf{w}_{2,k}^\top \phi_{1,n} + b_{2,k} \right)$$

- Finally, we can write

$$\mathbf{y}_n | \mathbf{X}, \mathbf{W}_1, \mathbf{b}_1, \mathbf{W}_2, \mathbf{b}_2, \mathbf{W}_3 \sim \mathcal{N}(\mathbf{W}_3^\top \phi_{2,n}, \tau^{-1} \mathbf{I}_D)$$

Section 4

Obtaining Model Uncertainty

4. Obtaining Model Uncertainty

- Recall that the **approximate predictive distribution:**

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

where $\boldsymbol{\omega} = \{\mathbf{W}_i\}_{i=1}^{L}$ is our set of random variables for a model with L layers.

4. Obtaining Model Uncertainty

- We sample T sets of vectors of realizations from the Bernoulli distribution $\{\mathbf{z}_1^t, \cdots, \mathbf{z}_L^t\}_{t=1}^T$ with $\mathbf{z}_i^t = [\mathbf{z}_{i,j}^t]_{j=1}^{K_i}$, giving $\{\mathbf{W}_1^t, \cdots, \mathbf{W}_L^t\}_{t=1}^T$.

- We estimate the first moment:

$$\mathbb{E}_{q(\mathbf{y}^*|\mathbf{x}^*)}[\mathbf{y}^*] = \int \mathbf{y}^* q(\mathbf{y}^*|\mathbf{x}^*) d\mathbf{y}^*$$
$$\approx \frac{1}{T} \sum_{i=1}^{T} \hat{\mathbf{y}}^*(\mathbf{x}^*, \mathbf{W}_1^t, \cdots, \mathbf{W}_L^t),$$

  called **MC dropout**.

4. Obtaining Model Uncertainty

- We estimate the second moment:

$$\mathbb{E}_{q(\mathbf{y}^*|\mathbf{x}^*)}[(\mathbf{y}^*)^\top(\mathbf{y}^*)]$$
$$\approx \tau^{-1}\mathbf{I}_D + \frac{1}{T}\sum_{t=1}^{T}\hat{\mathbf{y}}^*(\mathbf{x}^*, \mathbf{W}_1^t, \cdots, \mathbf{W}_L^t)^\top \hat{\mathbf{y}}^*(\mathbf{x}^*, \mathbf{W}_1^t, \cdots, \mathbf{W}_L^t)$$

- The model's predictive variance:

$$\mathrm{Var}_{q(\mathbf{y}^*|\mathbf{x}^*)}(\mathbf{y}^*)$$
$$\approx \tau^{-1}\mathbf{I}_D + \frac{1}{T}\sum_{t=1}^{T}\hat{\mathbf{y}}^*(\mathbf{x}^*, \mathbf{W}_1^t, \cdots, \mathbf{W}_L^t)^\top \hat{\mathbf{y}}^*(\mathbf{x}^*, \mathbf{W}_1^t, \cdots, \mathbf{W}_L^t)$$
$$- \mathbb{E}_{q(\mathbf{y}^*|\mathbf{x}^*)}[\mathbf{y}^*]^\top \mathbb{E}_{q(\mathbf{y}^*|\mathbf{x}^*)}[\mathbf{y}^*]$$

4. Obtaining Model Uncertainty

Theoretical background

- $\hat{\mathbf{y}}^*(\mathbf{x}^*, \mathbf{W}_1^t, \cdots, \mathbf{W}_L^t) = \mathbf{f}^{\boldsymbol{\omega}}(\mathbf{x}^*)$

The **first moment** can be estimated as follows:

Proposition (in thesis)

Given $p(\mathbf{y}^*|\mathbf{f}^{\boldsymbol{\omega}}(\mathbf{x}^*)) = \mathcal{N}(\mathbf{y}^*; \mathbf{f}^{\boldsymbol{\omega}}(\mathbf{x}^*), \tau^{-1}\mathbf{I})$ *for some* $\tau > 0$, $\mathbb{E}_{q(\mathbf{y}^*|\mathbf{x}^*)}[\mathbf{y}^*]$ *can be estimated with the unbiased estimator*

$$\tilde{\mathbb{E}}[\mathbf{y}^*] := \frac{1}{T}\sum_{t=1}^{T} \mathbf{f}^{\hat{\boldsymbol{\omega}}_t}(\mathbf{x}^*) \xrightarrow[T\to\infty]{} \mathbb{E}_{q(\mathbf{y}^*|\mathbf{x}^*)}[\mathbf{y}^*]$$

, with $\hat{\boldsymbol{\omega}}_t \sim q(\boldsymbol{\omega})$.

4. Obtaining Model Uncertainty

Theoretical background

We estimate the **second raw moment** (for regression) using the following proposition:

Proposition (in thesis)

Given $p(\mathbf{y}^*|\mathbf{f}^{\boldsymbol{\omega}}(\mathbf{x}^*)) = \mathcal{N}(\mathbf{y}^*; \mathbf{f}^{\boldsymbol{\omega}}(\mathbf{x}^*), \tau^{-1}\mathbf{I})$ *for some* $\tau > 0$, $\mathbb{E}_{q(\mathbf{y}^*|\mathbf{x}^*)}[(\mathbf{y}^*)^\top(\mathbf{y}^*)]$ *can be estimated with the unbiased estimator*

$$\tilde{\mathbb{E}}[(\mathbf{y}^*)^\top(\mathbf{y}^*)] := \tau^{-1}\mathbf{I} + \frac{1}{T}\sum_{t=1}^{T}\mathbf{f}^{\hat{\boldsymbol{\omega}}_t}(\mathbf{x}^*)^\top\mathbf{f}^{\hat{\boldsymbol{\omega}}_t}(\mathbf{x}^*)$$

$$\xrightarrow[T\to\infty]{} \mathbb{E}_{q(\mathbf{y}^*|\mathbf{x}^*)}[(\mathbf{y}^*)^\top(\mathbf{y}^*)]$$

with $\hat{\boldsymbol{\omega}}_t \sim q(\boldsymbol{\omega})$ *and* $\mathbf{y}^*, \mathbf{f}^{\hat{\boldsymbol{\omega}}_t}(\mathbf{x}^*)$ *row vectors (thus the sum is over the outer-products).*

4. Obtaining Model Uncertainty

- Note that the dropout NN model itself is not changed.
- As a result, this information can be used with existing NN models trained with dropout.

Section 5

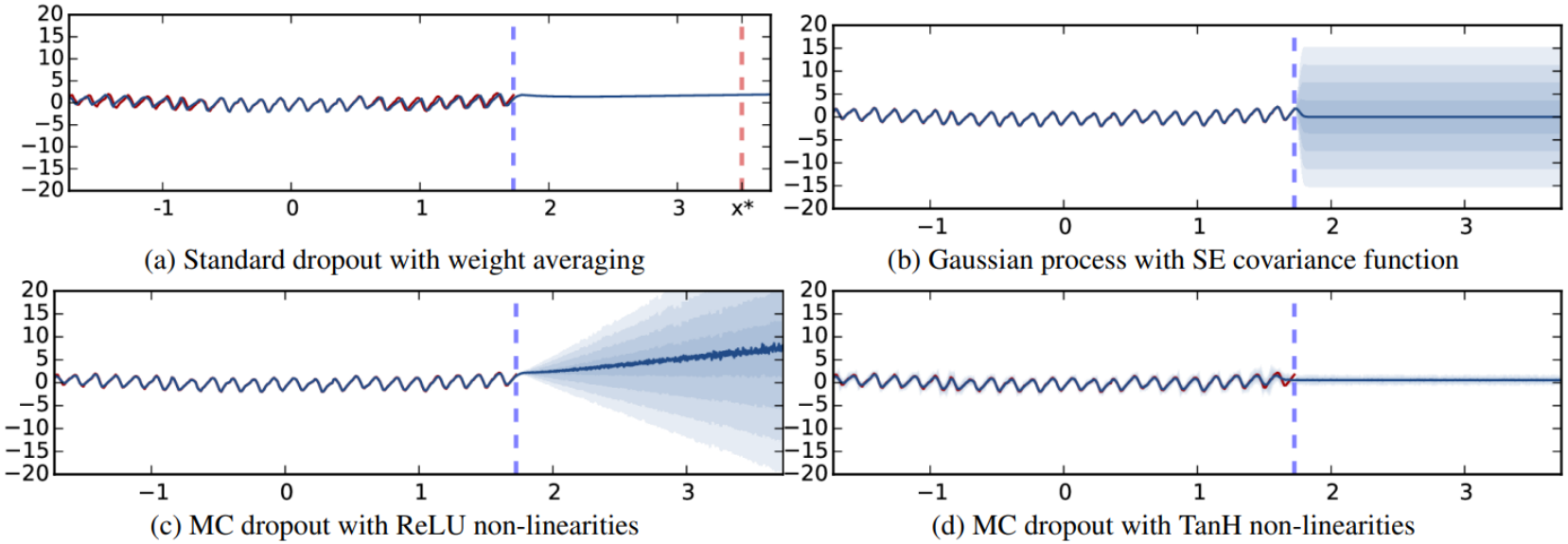
Experiments

5. Experiments

- We next perform an assessment of the properties of the uncertainty estimates obtained from dropout NNs
  - We compare the uncertainty obtained from different model architectures and non-linearities.
  - We show that model uncertainty is important for classification tasks using MNIST as an example.

5. Experiments

5.1 Model Uncertainty in Regression Tasks



(a) Standard dropout with weight averaging

(b) Gaussian process with SE covariance function

(c) MC dropout with ReLU non-linearities

(d) MC dropout with TanH non-linearities

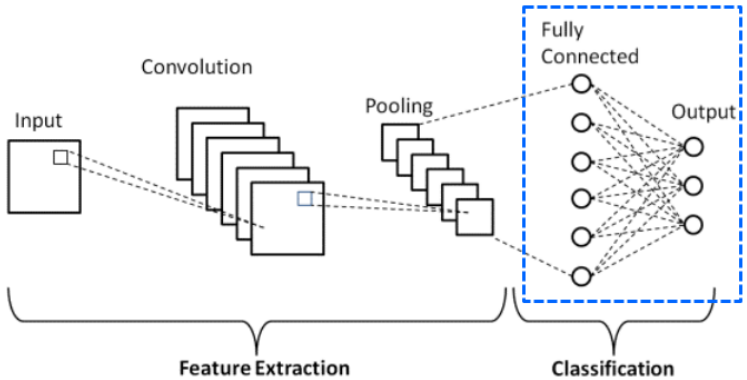
Figure 2. **Predictive mean and uncertainties on the Mauna Loa CO_2 concentrations dataset, for various models.** In red is the observed function (left of the dashed blue line); in blue is the predictive mean plus/minus two standard deviations (8 for fig. 2d). Different shades of blue represent half a standard deviation. Marked with a dashed red line is a point far away from the data: standard dropout confidently predicts an insensible value for the point; the other models predict insensible values as well but with the additional information that the models are uncertain about their predictions.

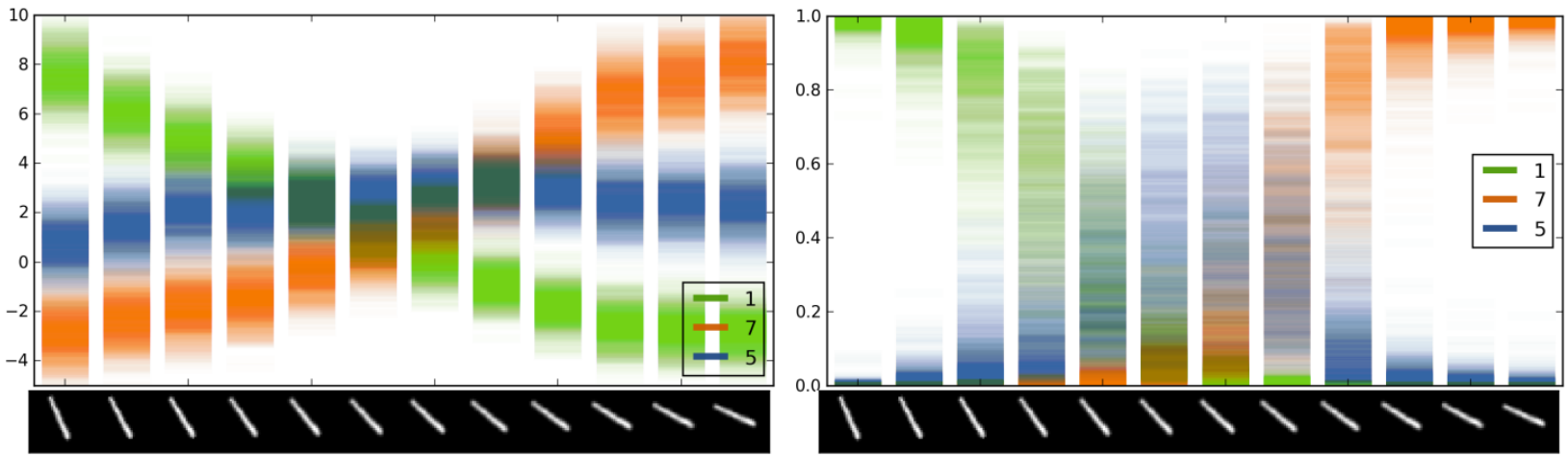
- Note that: Different covariance function s correspond to different uncertainty estimates.

$\Rightarrow$ ReLU and TanH approximate differnet GP covariance functions.

5. Experiments

5.2 Model Uncertainty in Classification Tasks





(a) Softmax *input* scatter

(b) Softmax *output* scatter

Figure 4. **A scatter of 100 forward passes of the softmax input and output for dropout LeNet.** On the X axis is a rotated image of the digit 1. The input is classified as digit 5 for images 6-7, even though model uncertainty is extremly large (best viewed in colour).

5. Experiments

5.2 Model Uncertainty in Classification Tasks

- If the uncertainty envelope of a class is far from that of other classes, then the input is classified with high confidence.
- On the other hand, if the uncertainty envelope intersects that of other classes, then even though the softmax output can be arbitrarily high, the softmax output uncertainty can be as large as the entire space.

References

- Yarin Gal, Dropout as a Bayesian Approximation: Appendix. 2016
- Yarin Gal, Uncertainty in Deep Learning. PhD Thesis, 2016
- C. E. Rasmussen and C. K. Williams, Gaussian processes for machine learning, volume 1. MIT press Cambridge, 2006.
- Tsuda, K., Kin, T., and Asai, K. (2002). Marginalized kernels for biological sequences. Bioinformatics.

감사합니다.