Dropout as a Bayesian Approximation: Insights and Applications

Yarin Gal and Zoubin Ghahramani

Discussion by: Chunyuan Li

Jan. 15, 2016

Main idea

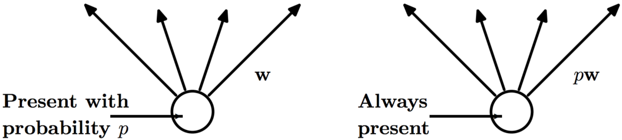
- In the framework of variational inference, the authors show that the **standard algorithm of SGD training with Dropout** is ensentially **optimizing the stochastic lower bound of Gaussian Processes whose kernel takes the form of neural networks**.

Outline

Dropout

Procedure

- Training stage: A unit is present with probability p
- Testing stage: The unit is always present and the weights are multiplied by p



(a) At training time (b) At test time

Intuition

- Training a neural network with dropout can be seen as training a collection of 2^K thinned networks with extensive weight sharing
- A single neural network to approximate averaging output at test time

Dropout for one-hidden-layer Neural Networks

- Dropout local units*

$$\hat{\boldsymbol{y}} = \big(g((\boldsymbol{x} \odot \mathbf{b}_1)\mathbf{W}_1) \odot \mathbf{b}_2\big)\mathbf{W}_2 \tag{1}$$

  - Input $\boldsymbol{x}$ and Output $\boldsymbol{y}$;
  - g: activation function; Weights: $\mathbf{W} \in \mathbb{R}^{K_{\ell-1} \times K_\ell}$
  - $\mathbf{b}_\ell$ is binary dropout vairiables

- Equivalent to multiplying the global weight matrices by the binary vectors to dropout entire rows:

$$\hat{\boldsymbol{y}} = g(\boldsymbol{x}(\mathrm{diag}(\mathbf{b}_1)\mathbf{W}_1))(\mathrm{diag}(\mathbf{b}_2)\mathbf{W}_2) \tag{2}$$

- Application to regression

$$\mathcal{L} = \frac{1}{2N}\sum_{n=1}^{N} \|\boldsymbol{y}_n - \hat{\boldsymbol{y}}_n\|_2^2 + \lambda(\|\mathbf{W}_1\|^2 + \|\mathbf{W}_2\|^2) \tag{3}$$

*bias term is ignored for simplicity

Gaussian Processes

- $\boldsymbol{f}$ is the GP function
  $\underbrace{p(\boldsymbol{f}|\mathbf{X},\mathbf{Y})}_{\text{Posterior}} \propto \underbrace{p(\boldsymbol{f})}_{\text{Prior}} \underbrace{p(\mathbf{Y}|\mathbf{X},\boldsymbol{f})}_{\text{Likelihood}}$
- Applications
  - Regression
    $\mathbf{F}|\mathbf{X} \sim \mathcal{N}(0, \mathbf{K}(\mathbf{X},\mathbf{X})), \quad \mathbf{Y}|\mathbf{F} \sim \mathcal{N}(\mathbf{F}, \tau^{-1}\mathbf{I}_N)$

Variational Inference

- The predictive distribution

$$\mathbf{K}(\boldsymbol{y}^*|\boldsymbol{x}^*, \mathbf{X}, \mathbf{Y}) = \int p(\boldsymbol{y}^*|\boldsymbol{x}^*, \boldsymbol{w}) \underbrace{p(\boldsymbol{w}|\mathbf{X}, \mathbf{Y})}_{\approx q(\boldsymbol{w})} \mathrm{d}\boldsymbol{w} \qquad (4)$$

  - Objective: $\text{argmin}_q \, \mathrm{KL}\big(q(\boldsymbol{w})|p(\boldsymbol{w}|\mathbf{X}, \mathbf{Y})\big)$
  - Variational Prediction: $q(\boldsymbol{y}^*|\boldsymbol{x}^*) = \int p(\boldsymbol{y}^*|\boldsymbol{x}^*, \boldsymbol{w})q(\boldsymbol{w})\mathrm{d}\boldsymbol{w}$
- Log evidence lower bound

$$\mathcal{L}_{\text{VI}} = \int q(\boldsymbol{w}) \log p(\mathbf{Y}|\mathbf{X}, \boldsymbol{w})\mathrm{d}\boldsymbol{w} - \mathrm{KL}\big(q(\boldsymbol{w})||p(\boldsymbol{w})\big) \qquad (5)$$

  - Objective: $\arg\max_q \mathcal{L}_{\text{VI}}$
  - A variational distribution $q(\boldsymbol{w})$ that explains the data well while still being close to prior

A single-layer neural network example

- Setup
  - Q: input dimension,
    K: number of hidden units,
    D: ouput dimesnion
  - Goal: Learn $\mathbf{W}_1 \in \mathbb{R}^{Q \times K}$ and $\mathbf{W}_2 \in \mathbb{R}^{K \times D}$ to map $\mathbf{X} \in \mathbb{R}^{N \times Q}$ to $\mathbf{Y} \in \mathbb{R}^{N \times D}$
- Idea: Introduce $\mathbf{W}_1$ and $\mathbf{W}_2$ to GP approxiamtion

Introduce $\mathbf{W}_1$

- Define the kernel of GP

$$\mathbf{K}(\boldsymbol{x}, \boldsymbol{x}') = \int p(\boldsymbol{w}) g(\boldsymbol{w}^\top \boldsymbol{x}) g(\boldsymbol{w}^\top \boldsymbol{x}') \mathrm{d}\boldsymbol{w} \tag{6}$$

- Resort to Monte Carlo integration, with generative process:

$$\boxed{\boldsymbol{w}_k \sim p(\boldsymbol{w}),\ \mathbf{W}_1 = [\boldsymbol{w}_k]_{k=1}^K,}\ \widehat{\mathbf{K}}(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{K} \sum_{k=1}^{K} g(\boldsymbol{w}_k^\top \boldsymbol{x}) g(\boldsymbol{w}_k^\top \boldsymbol{x}')$$

$$\mathbf{F} | \mathbf{X}, \mathbf{W}_1 \sim \mathcal{N}(\mathbf{0}, \widehat{\mathbf{K}}),\ \mathbf{Y} | \mathbf{F} \sim \mathcal{N}(\mathbf{F}, \tau^{-1} \mathbf{I}_N) \tag{7}$$

where K is the number of hidden units

Introduce $\mathbf{W}_2$

- Analytically integrating wrt $\mathbf{F}$
  The predictive distribution

$$p(\mathbf{Y}|\mathbf{X}) = \int p(\mathbf{Y}|\mathbf{F})p(\mathbf{F}|\mathbf{X}, \mathbf{W}_1)p(\mathbf{W}_1)\mathrm{d}\boldsymbol{w}_1\mathrm{d}\boldsymbol{f} \tag{8}$$

  can be rewritten as

$$p(\mathbf{Y}|\mathbf{X}) = \int \mathcal{N}(\mathbf{Y}; \mathbf{0}, \mathbf{\Phi}\mathbf{\Phi}^\top + \tau^{-1}\mathbf{I}_N)p(\mathbf{W}_1)\mathrm{d}\boldsymbol{w}_1 \tag{9}$$

  where $\mathbf{\Phi} = [\boldsymbol{\phi}]_{n=1}^N$, $\boldsymbol{\phi} = \sqrt{\frac{1}{K} g(\mathbf{W}_1^\top \boldsymbol{x})}$

- For $\boxed{\mathbf{W}_2 = [\boldsymbol{w}_2]_{d=1}^D,\ \boldsymbol{w}_d \sim \mathcal{N}(0, \mathbf{I}_K)}$

$$\mathcal{N}(\boldsymbol{y}_d; \mathbf{0}, \mathbf{\Phi}\mathbf{\Phi}^\top + \tau^{-1}\mathbf{I}_N) = \int \mathcal{N}(\boldsymbol{y}_d; \mathbf{\Phi}\boldsymbol{w}_d, \tau^{-1}\mathbf{I}_N)\mathcal{N}(\boldsymbol{w}_d; 0, \mathbf{I}_K)\mathrm{d}\boldsymbol{w}_1$$

$$p(\mathbf{Y}|\mathbf{X}) = \int p(\mathbf{Y}|\mathbf{X}, \mathbf{W}_1, \mathbf{W}_2)p(\mathbf{W}_1)p(\mathbf{W}_2)\mathrm{d}\boldsymbol{w}_1\mathrm{d}\boldsymbol{w}_2 \tag{10}$$

Variational Inference in the Approximate Model

$$q(\mathbf{W}_1, \mathbf{W}_2) = q(\mathbf{W}_1)q(\mathbf{W}_2) \tag{11}$$

- To mimic Dropout, $q(\mathbf{W}_1)$ is factorised over input dimension, each of them is a Gaussian mixture distribution with two components,

$$q(\mathbf{W}_1) = \prod_{q=1}^{Q} q(\boldsymbol{w}_q),\ q(\boldsymbol{w}_q) = p_1 \mathcal{N}(\boldsymbol{m}_q, \sigma^2 \mathbf{I}_K) + (1 - p_1)\mathcal{N}(0, \sigma^2 \mathbf{I}_K) \tag{12}$$

with $p_1 \in [0, 1]$, $\boldsymbol{m}_q \in \mathbb{R}^K$

- The same for $q(\mathbf{W}_2)$

$$q(\mathbf{W}_2) = \prod_{k=1}^{K} q(\boldsymbol{w}_k),\ q(\boldsymbol{w}_k) = p_2 \mathcal{N}(\boldsymbol{m}_k, \sigma^2 \mathbf{I}_D) + (1 - p_2)\mathcal{N}(0, \sigma^2 \mathbf{I}_D) \tag{13}$$

with $p_1 \in [0, 1]$, $\boldsymbol{m}_q \in \mathbb{R}^D$

- Optimise over parameters, especially $M_1 = [\boldsymbol{m}_q]_{q=1}^{Q}$, $\mathbf{M}_2 = [\boldsymbol{m}_k]_{k=1}^{K}$

Evaluating the Log Evidence Lower Bound for Regression

- Log evidence lower bound

$$\mathcal{L}_{\text{GP-VI}} = \underbrace{\int q(\mathbf{W}_1, \mathbf{W}_2) \log p(\mathbf{Y}|\mathbf{X}, \mathbf{W}_1, \mathbf{W}_2) \mathrm{d}\mathbf{W}_1 \mathrm{d}\mathbf{W}_2}_{\mathcal{L}_1} - \underbrace{\mathrm{KL}\big(q(\mathbf{W}_1, \mathbf{W}_2) || p(\mathbf{W}_1, \mathbf{W}_2)\big)}_{\mathcal{L}_2} \tag{14}$$

- Approximation of $\mathcal{L}_2^{\dagger}$
  - For large enough K we can approximate the KL divergence term as

$$\mathrm{KL}(q(\mathbf{W}_1)||p(\mathbf{W}_1)) \approx QK(\sigma^2 - \log(\sigma^2) - 1) + \boxed{\frac{p_1}{2} \sum_{q=1}^{Q} \boldsymbol{m}_q^\top \boldsymbol{m}_q} \tag{15}$$

  - Similarly for $\mathrm{KL}(q(\mathbf{W}_2)||p(\mathbf{W}_2))$

† Following Proposition 1 on KL of a Mixture of Gaussians in Appendix

$\mathcal{L}_1$: Monte Carlo integration

- Parameterization

$$\mathcal{L}_1 = q(\mathbf{b}_1, \epsilon_1, \mathbf{b}_2, \epsilon_2) \log p(\mathbf{Y}|\mathbf{X}, \mathbf{W}_1(\mathbf{b}_1, \epsilon_1), \mathbf{W}_2(\mathbf{b}_2, \epsilon_2)) \mathrm{d}\mathbf{b}_1 \mathrm{d}\mathbf{b}_2 \mathrm{d}\epsilon_1 \mathrm{d}\epsilon_2 \tag{16}$$

$$\begin{aligned} \mathbf{W}_1 &= \mathrm{diag}(\mathbf{b}_1)(\mathbf{M}_1 + \sigma\boldsymbol{\epsilon}_1) + (1 - \mathrm{diag}(\mathbf{b}_1))\sigma\boldsymbol{\epsilon}_1 \\ \mathbf{W}_2 &= \mathrm{diag}(\mathbf{b}_2)(\mathbf{M}_2 + \sigma\boldsymbol{\epsilon}_2) + (1 - \mathrm{diag}(\mathbf{b}_2))\sigma\boldsymbol{\epsilon}_2 \end{aligned} \tag{17}$$

where

$$\begin{aligned} &\boldsymbol{\epsilon}_1 \sim \mathcal{N}(\mathbf{0}, I_{Q\times K}), \mathrm{b}_{1q} \sim \mathrm{Bernoulli}(p_1), \\ &\boldsymbol{\epsilon}_2 \sim \mathcal{N}(\mathbf{0}, I_{K\times D}), \mathrm{b}_{2k} \sim \mathrm{Bernoulli}(p_2), \end{aligned} \tag{18}$$

- Take a single sample

$$\mathcal{L}_{\text{GP-MC}} = \underbrace{\log p(\mathbf{Y}|\mathbf{X}, \widehat{\mathbf{W}}_1, \widehat{\mathbf{W}}_2)}_{\mathcal{L}_{\text{1-MC}}} - \mathrm{KL}\big(q(\mathbf{W}_1, \mathbf{W}_2)||p(\mathbf{W}_1, \mathbf{W}_2)\big) \tag{19}$$

Optimising $\mathcal{L}_{\text{GP-MC}}$ converges to the same limit as $\mathcal{L}_{\text{GP-VI}}$.

$\mathcal{L}_1$: Monte Carlo integration

- Regression

$$\begin{aligned}\mathcal{L}_{1\text{-MC}} &= \log p(\mathbf{Y}|\mathbf{X}, \widehat{\mathbf{W}}_1, \widehat{\mathbf{W}}_2) \\ &= \sum_{d=1}^{D} \log \mathcal{N}(\boldsymbol{y}_d; \boldsymbol{\phi}\hat{\boldsymbol{w}}_d, \tau^{-1}\mathbf{I}_N) \\ &= -\frac{ND}{2}\log(2\pi) + \frac{ND}{2}\log(\tau) - \sum_{d=1}^{D}\frac{\tau}{2}\|\boldsymbol{y}_d - \boldsymbol{\phi}\hat{\boldsymbol{w}}_d\|_2^2 \end{aligned} \tag{20}$$

- Sum over the rows instead of the columns of $\widehat{\mathbf{Y}}$

$$\sum_{d=1}^{D}\frac{\tau}{2}\|\boldsymbol{y}_d - \boldsymbol{\phi}\hat{\boldsymbol{w}}_d\|_2^2 = \boxed{\sum_{n=1}^{N}\frac{\tau}{2}\|\boldsymbol{y}_n - \boldsymbol{\phi}\hat{\boldsymbol{w}}_n\|_2^2} \tag{21}$$

where $\hat{\boldsymbol{y}}_n = \boldsymbol{\phi}\widehat{\mathbf{W}}_2 = \sqrt{\frac{1}{K}}g(\boldsymbol{x}_n\widehat{\mathbf{W}}_1)\widehat{\mathbf{W}}_2$

Recover SGD training with Dropout

- Take the approximation for $\mathcal{L}_1$ and $\mathcal{L}_2$ for the bound, and ignoring constant terms τ and σ

$$\mathcal{L}_{\text{GP-MC}} = -\frac{\tau}{2}\sum_{n=1}^{N} \|\boldsymbol{y}_n - \hat{\boldsymbol{y}}_n\|_2^2 - \frac{p_1}{2}\|\mathbf{M}_1\|_2^2 - \frac{p_2}{2}\|\mathbf{M}_2\|_2^2 \tag{22}$$

- Setting σ tend to 0

$$\mathbf{W}_1 = \text{diag}(\mathbf{b}_1)(\mathbf{M}_1 + \sigma\boldsymbol{\epsilon}_1) + (1 - \text{diag}(\mathbf{b}_1))\sigma\boldsymbol{\epsilon}_1 \Rightarrow$$

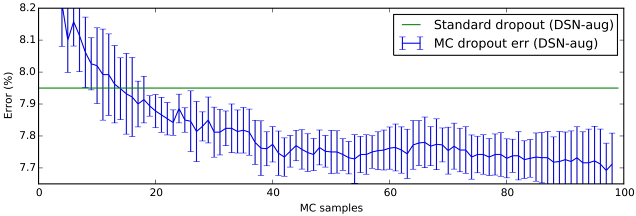
$$\widehat{\mathbf{W}}_1 \approx \text{diag}(\hat{\mathbf{b}}_1)\mathbf{M}_1,\ \widehat{\mathbf{W}}_2 \approx \text{diag}(\hat{\mathbf{b}}_2)\mathbf{M}_2 \tag{23}$$

$$\hat{\boldsymbol{y}}_n = \sqrt{\frac{1}{K}}g(\boldsymbol{x}_n\widehat{\mathbf{W}}_1)\widehat{\mathbf{W}}_2 = \sqrt{\frac{1}{K}}g(\boldsymbol{x}_n(\text{diag}(\hat{\mathbf{b}}_1)\mathbf{M}_1))(\text{diag}(\hat{\mathbf{b}}_2)\mathbf{M}_2) \tag{24}$$

More Applications

- Convolutional Neural Networks

  Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference, arXiv:1506.02158, 2015



- Recurrent Neural Networks

  A Theoretically Grounded Application of Dropout in Recurrent Neural Networks, arXiv:1512.05287, 2015
- Reinforcement Learning

  Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning, arXiv:1506.02142, 2015