Bayesian Deep Learning

Kangcheng Hou

Zhejiang University

May 19, 2018

Bayesian framework

Bayes theorem

$$\text{posterior} \propto \text{likelihood} \times \text{prior}$$

$$p(\mathbf{w}|\mathbf{X},\mathbf{Y}) = \frac{p(\mathbf{Y}|\mathbf{X},\mathbf{w})p(\mathbf{w})}{p(\mathbf{Y}|\mathbf{X})}$$

Having the information of posterior distribution, we can predict an output for a new input point $\mathbf{x}^*$

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

An important component in Bayesian framework is model evidence,

$$p(\mathbf{Y}|\mathbf{X},\mathcal{H}) = \int p(\mathbf{Y}|\mathbf{X},\mathbf{w},\mathcal{H})p(\mathbf{w}|\mathcal{H})d\mathbf{w}$$

It can be seen as marginalising the likelihood over $\mathbf{w}$.

Variational Inference

We are interested in the posterior distribution $p(\mathbf{w}|\mathbf{X}, \mathbf{Y})$, but this cannot usually be evaluated analytically.

Bayesian Deep Learning I

Our target when developing is to make as less change to the current Deep Learning structure as possible and get uncertainty information from the model. An important quantity in BDL is the posterior distribution of the weights.

$$p(\mathbf{w}|\mathbf{X},\mathbf{Y})$$

Using variational inference, we use a parametrized distribution $q_\theta(\mathbf{w})$ to approximate the posterior distribution $p(\mathbf{w}|\mathbf{X},\mathbf{Y})$. We minimizes the KL divergence

$$\begin{aligned}\mathsf{KL}(q_\theta(\mathbf{w})||p(\mathbf{w}|\mathbf{X},\mathbf{Y})) &= \int q_\theta(\mathbf{w}) \ln \frac{q_\theta(\mathbf{w})}{p(\mathbf{w}|\mathbf{X},\mathbf{Y})} d\mathbf{w} \\ &\propto \int q_\theta(\mathbf{w}) \ln \frac{q_\theta(\mathbf{w})}{p(\mathbf{w})} - \int q_\theta(\mathbf{w}) \ln p(\mathbf{Y}|\mathbf{X},\mathbf{w}) d\mathbf{w} \\ &= \mathsf{KL}(q_\theta(\mathbf{w})||p(\mathbf{w})) - \int q_\theta(\mathbf{w}) \ln p(\mathbf{Y}|\mathbf{X},\mathbf{w}) d\mathbf{w}\end{aligned}$$

Bayesian Deep Learning II

In DL context, this means the objective function is

$$\mathcal{L}_{\mathsf{VI}}(\theta) = -\sum_{i=1}^{N} \int q_\theta(\mathbf{w}) \ln(p(\mathbf{y}_i | f^{\mathbf{w}}(\mathbf{x}_i))) d\mathbf{w} + \mathsf{KL}(q_\theta(\mathbf{w}) || p(\mathbf{w}))$$

This objective function is not scalable because first the summed-over term is not tractable for DL model and N terms is large in big data century. Using stochastic variational inference will give help with the large N problem. And the monte carlo integration method will help us cope with the

$$\int q_\theta(\mathbf{w}) \ln p(\mathbf{y}_i | f^{\mathbf{w}}(\mathbf{x}_i)) d\mathbf{w}$$

term.

Monte Carlo estimation for VI I

Now the problem is evaluating

$$\int q_\theta(\mathbf{w}) \ln p(\mathbf{y}_i | f^{\mathbf{w}}(\mathbf{x}_i)) d\mathbf{w}$$

and optimize the quantity w.r.t θ. We consider a easier from of task

$$I(\theta) = \frac{\partial}{\partial \theta} \int f(x) p_\theta(x) dx$$

Here we use $p_\theta(x) = \mathcal{N}(x; \mu, \sigma^2)$ and $f(x)$ can be arbitrary. We introduce three methods:

Monte Carlo estimation for VI II

The score function estimator

$$\begin{aligned}\frac{\partial}{\partial\theta}\int f(x)p_\theta(x)dx &= \int f(x)\frac{\partial}{\partial\theta}p_\theta(x)dx \\ &= \int f(x)\frac{\partial}{\partial\theta}\log p_\theta(x)p_\theta(x)dx\end{aligned}$$

To estimate this quantity, we estimate

$$\mathbb{E}_{x\sim p_\theta(x)}[f(x)\frac{\partial}{\partial\theta}\ln p_\theta](x)$$

Monte Carlo estimation for VI III

Reparametrisation trick

If we can reparametrize the $p_\theta(x)$, for example, $p_\theta(x)$ can be $\mathcal{N}(\mu, \theta)$. We can thus reparametrize the $p_\theta(x)$ as $g(\theta, \epsilon) = \mu + \sigma\epsilon$ with $p(\epsilon) = \mathcal{N}(\epsilon; 0, I)$. We have the property that

$$p_\theta(x)dx = p(\epsilon)d\epsilon$$

$$\begin{aligned}
\nabla_\theta \int f(x)p_\theta(x)dx &= \nabla_\theta \int f(x)p(\epsilon)d\epsilon \\
&= \nabla_\theta \int f(g(\theta, \epsilon))p(\epsilon)d\epsilon \\
&= \mathbb{E}_{\epsilon\sim p(\epsilon)}[f(g(\theta, \epsilon))] \\
&= \mathbb{E}_{\epsilon\sim p(\epsilon)}[\nabla_\theta f(g(\theta, \epsilon))] \\
&= \mathbb{E}_{\epsilon\sim p(\epsilon)}[f'(g(\theta, \epsilon))\nabla_\theta g(\theta, \epsilon)]
\end{aligned}$$

Monte Carlo estimation for VI IV

Will not introduce the third method here.

Practical inference in Bayesian neural networks I

Previous work use fully fatorised Gaussian approximating distributions and characteristic function estimator in the approximation. This work will rely on the pathwise derivative estimator instead, so can make use of more interesting non-Gaussian approximating distributions. And to avoid losing weight correlations, fatorise the distribution for each weight row $\mathbf{w}_{l,i}$ in each weight matrix $\mathbf{W}_l$. With these two improvements, we can see that it is closely tied to SRT.

Practical inference in Bayesian neural networks II

Note that we are going to optimize the quantity

$$\mathcal{L}_{\mathsf{VI}}(\theta) = -\sum_{i=1}^{N} \int q_\theta(\mathbf{w}) \ln(p(\mathbf{y}_i | f^{\mathbf{w}}(\mathbf{x}_i)))d\mathbf{w} + \mathsf{KL}(q_\theta(\mathbf{w})||p(\mathbf{w}))$$

which can be optimized using batch information

$$\hat{\mathcal{L}}_{\mathsf{VI}}(\theta) = -\frac{N}{M} \sum_{i \in S} \int q_\theta(\mathbf{w}) \ln(p(\mathbf{y}_i | f^{\mathbf{w}}(\mathbf{x}_i)))d\mathbf{w} + \mathsf{KL}(q_\theta(\mathbf{w})||p(\mathbf{w}))$$

We reparametrize each $q_{\theta_{l,i}}(\mathbf{w}_{l,i})$ as $\boldsymbol{w}_{l,i} = g(\theta_{l,i}, \epsilon_{l,i})$ with some specified $p(\epsilon_{l,i})$. We write $p(\boldsymbol{\epsilon}) = \prod_{l,i}(\epsilon_{l,i})$ and $\mathbf{w} = g(\theta, \boldsymbol{\epsilon})$.

Practical inference in Bayesian neural networks III

Thus our objective function is

$$\begin{aligned}\hat{\mathcal{L}}_{\mathsf{VI}}(\theta) &= -\frac{N}{M}\sum_{i\in S}\int q_\theta(\mathbf{w})\ln p(\mathbf{y}_i|f^{\mathbf{w}}(\mathbf{x}_i))d\mathbf{w} + \mathsf{KL}(q_\theta(\mathbf{w})||p(\mathbf{w}))\\ &= -\frac{N}{M}\sum_{i\in S}\int p(\boldsymbol{\epsilon})\ln p(\mathbf{y}_i|f^{g(\theta,\boldsymbol{\epsilon})}(\mathbf{x}_i))d\boldsymbol{\epsilon} + \mathsf{KL}(q_\theta(\mathbf{w})||p(\mathbf{w}))\end{aligned}$$

Replace the expected log likelihood term with its stochastic estimator, we get the MC estimator:

$$\hat{\mathcal{L}}_{\mathsf{MC}}(\theta) = -\frac{N}{M}\sum_{i\in S}\ln p(\mathbf{y}_i|f^{g(\theta,\boldsymbol{\epsilon})}(\mathbf{x}_i)) + \mathsf{KL}(q_\theta(\mathbf{w})||p(\mathbf{w}))$$

And

$$\mathbb{E}_{S,\boldsymbol{\epsilon}}[\hat{\mathcal{L}}_{\mathsf{MC}}] = \mathcal{L}_{\mathsf{VI}}$$

Optimizing $\hat{\mathcal{L}}_{\mathsf{MC}}$ w.r.t θ would converge to the same optima as optimizing $\mathcal{L}_{\mathsf{VI}}$.

Practical inference in Bayesian neural networks IV

To make predictive distribution, we do the following approximation:

$$
\begin{aligned}
p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) &= \int p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w}) p(\mathbf{w}|\mathbf{X}, \mathbf{Y}) d\mathbf{w} \\
&\approx \int p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w}) q_\theta(\mathbf{w}) d\mathbf{w} \\
&\approx \frac{1}{T} \sum_{t=1}^{T} p(\mathbf{y}^*|\mathbf{x}^*, \hat{\mathbf{w}}_t) \qquad \hat{\mathbf{w}}_t \sim q_\theta(\mathbf{w})
\end{aligned}
$$

Stochastic regularisation techniques I

Stochastic regularization techniques are techniques used to regularize deep learning models through the injection of stochastic noise into the model. We will try to derive the fact that dropout is equivalent to doing variational inference on the neural net.
We will introduce a simple situation where the output

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

For every forward pass in neural net, we sample random vector of dimension the same of input $\mathbf{x}$ and latent variable $\sigma(\mathbf{x}\mathbf{M}_1 + \mathbf{b})$, $\boldsymbol{\epsilon}_1$ and $\boldsymbol{\epsilon}_2$ respectively. Then we do element wise product to the corresponding variable. So the output becomes

$$\hat{\mathbf{y}} = (\sigma((\mathbf{x} \odot \hat{\boldsymbol{\epsilon}_1})\mathbf{M}_1 + \mathbf{b}) \odot \hat{\boldsymbol{\epsilon}_2})\mathbf{M}_2$$

Stochastic regularisation techniques II

From above, it seems that noise were applied to the feature space, say, $\mathbf{x}, \mathbf{h}$. But we can transform the expression such that the noise were applied to the weight space as follows:

$$\begin{aligned}\hat{\mathbf{y}} &= \hat{\mathbf{h}}\mathbf{M}_2 \\ &= (\mathbf{h} \odot \hat{\boldsymbol{\epsilon}}_2)\mathbf{M}_2 \\ &= (\mathbf{h} \cdot \text{diag}(\hat{\boldsymbol{\epsilon}}_2))\mathbf{M}_2 \\ &= \mathbf{h} \cdot (\text{diag}(\hat{\boldsymbol{\epsilon}}_2)\mathbf{M}_2) \\ &= \sigma(\mathbf{x}(\text{diag}(\hat{\boldsymbol{\epsilon}}_1)\mathbf{M}_1) + \mathbf{b})(\text{diag}(\hat{\boldsymbol{\epsilon}}_2)\mathbf{M}_2)\end{aligned}$$

With $\text{diag}(\hat{\boldsymbol{\epsilon}}_1)\mathbf{M}_1$ as $\hat{\mathbf{W}}_1$ and $\text{diag}(\hat{\boldsymbol{\epsilon}}_2)\mathbf{M}_2$ as $\hat{\mathbf{W}}_2$ We can write

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

It can be seen as the randomized version of normal forward propagation

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

Stochastic regularisation techniques III

Now we show the relation between **dropout** and **variational inference**. We first note that we want to optimize the target of

$$\mathcal{L}(\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}) := E^{\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}}(\mathbf{X}, \mathbf{Y}) + \lambda_1 ||\mathbf{W}_1||^2 + \lambda_2 ||\mathbf{W}_2||^2 + \lambda_3 ||\mathbf{b}||^2$$

In dropout scenerio, it will be

$$\begin{aligned} \hat{\mathcal{L}}_{\text{dropout}}(\mathbf{M}_1, \mathbf{M}_2, \mathbf{b}) := & \frac{1}{M} \sum_{i \in S} E^{\hat{\mathbf{W}}_1^i, \hat{\mathbf{W}}_2^i, \mathbf{b}}(\mathbf{x}_i, \mathbf{y}_i) \\ & + \lambda_1 ||\mathbf{M}_1||^2 + \lambda_2 ||\mathbf{M}_2||^2 + \lambda_3 ||\mathbf{b}||^2 \end{aligned}$$

It can be shown that

$$\begin{aligned} E^{\mathbf{M}_1, \mathbf{M}_2, \mathbf{b}}(\mathbf{x}, \mathbf{y}) &= \frac{1}{2} ||\mathbf{y} - \mathbf{f}^{\mathbf{M}_1, \mathbf{M}_2, \mathbf{b}}(\mathbf{x})||^2 \\ &= -\frac{1}{\tau} \log p(\mathbf{y} | \mathbf{f}^{\mathbf{M}_1, \mathbf{M}_2, \mathbf{b}}(\mathbf{x})) + \text{const} \end{aligned}$$

where

$$p(\mathbf{y} | \mathbf{f}^{\mathbf{M}_1, \mathbf{M}_2, \mathbf{b}}(\mathbf{x})) = \mathcal{N}(\mathbf{y}; \mathbf{f}^{\mathbf{M}_1, \mathbf{M}_2, \mathbf{b}}(\mathbf{x}), \tau^{-1} I)$$

Stochastic regularisation techniques IV

Plug the quantity into the dropout loss expression,

$$\hat{\mathcal{L}}_{\mathsf{dropout}}(\mathbf{M}_1, \mathbf{M}_2, \mathbf{b}) := -\frac{1}{M\tau}\sum_{i\in S}\log p(\mathbf{y}|\mathbf{f}^{g(\theta;\hat{\epsilon_i})}(\mathbf{x}_i)) \\ + \lambda_1||\mathbf{M}_1||^2 + \lambda_2||\mathbf{M}_2||^2 + \lambda_3||\mathbf{b}||^2$$

Here

$$\{\hat{\mathbf{W}}_1^i, \hat{\mathbf{W}}_2^i, \mathbf{b}\} =: g(\theta, \hat{\boldsymbol{\epsilon}}_i)$$

Recall that the MC estimator for the loss is

$$\hat{\mathcal{L}}_{\mathsf{MC}}(\theta) = -\frac{N}{M}\sum_{i\in S}\ln p(\mathbf{y}_i|f^{g(\theta,\boldsymbol{\epsilon})}(\mathbf{x}_i)) + \mathsf{KL}(q_\theta(\mathbf{w})||p(\mathbf{w}))$$

The differences are the normalizing constant and the KL divergence. If we define the prior $p(\mathbf{w})$ s.t. the following holds:

$$\frac{\partial}{\partial\theta}KL(q_\theta(\mathbf{w})||p(\mathbf{w})) = \frac{\partial}{\partial\theta}N\tau(\lambda_1||\mathbf{M}_1||^2 + \lambda_2||\mathbf{M}_2||^2 + \lambda_3||\mathbf{b}||^2)$$

Then the two algorithm will be the same.

Stochastic regularisation techniques V

So what have we found just now? We showed that the dropout NN is equivalent to posterior distribution reparametrized as follows

$$g(\theta, \hat{\boldsymbol{\epsilon}}_i) := \{\hat{\mathbf{W}}_1^i, \hat{\mathbf{W}}_2^i, \mathbf{b}\}$$

where

$$\hat{\boldsymbol{\epsilon}}_i \sim \text{Bernolli}(p_i)$$

With this approximated posterior distribution, we can optimize w.r.t θ to get an optimal approximated posterior distribution.

KL condition I

For VI to result in an same optimization as dropout NN, the following KL condition has to be satisfied.

$$\frac{\partial}{\partial \theta} KL(q_\theta(\mathbf{w})||p(\mathbf{w})) = \frac{\partial}{\partial \theta} N\tau(\lambda_1||\mathbf{M}_1||^2 + \lambda_2||\mathbf{M}_2||^2 + \lambda_3||\mathbf{b}||^2)$$

Solving this should not be an easy problem. We need to choose $q_\theta(\mathbf{w})$ and $p(\mathbf{w})$ properly to achieve this KL condition. For example, it can be shown that setting the model prior to

$$p(\mathbf{w}) = \prod_{i=1}^{L} p(\mathbf{W}_i) = \prod_{i=1}^{L} \mathcal{N}(0, \mathbf{I}/l_i^2)$$

in other words independent normal priors over each weight, with prior length-scale

$$l_i^2 = \frac{2N\tau\lambda_i}{1-p_i}$$

KL condition II

the KL condition approximately holds for a large enough number of hidden units and a bernoulli variational distribution.

What is a prior length-scale?

Model uncertainty in Bayesian neural networks I

We now show that the model uncertainty can be obtained from dropout NN models. Our approximate predictive distribution is given by

$$q_\theta^*(\mathbf{y}^*|\mathbf{x}^*) = \int p(\mathbf{y}^*|f^{\mathbf{w}}(\mathbf{x}^*))q_\theta^*(\mathbf{w})d\mathbf{w}$$

where $q_\theta^*(\mathbf{w})$ is given by variational inference. It can be shown that given that the likelihood is Gaussian,

$$p(\mathbf{y}^*|\mathbf{f}^{\mathbf{w}}(\mathbf{x}^*)) = \mathcal{N}(\mathbf{y}^*; \mathbf{f}^{\mathbf{w}}(\mathbf{x}^*), \tau^{-1}\mathbf{I})$$

Then

$$\frac{1}{T}\sum_{t=1}^{T} \mathbf{f}^{\hat{\mathbf{w}}_t}(\mathbf{x}^*) \xrightarrow{T\to\infty} \mathbb{E}_{q_\theta^*(\mathbf{y}^*|\mathbf{x}^*)}[\mathbf{y}^*]$$

This means we do forward pass T times and get the average, and if $T \to \infty$, then the mean of the samples will be the same as that of predictive distribution.

Model uncertainty in Bayesian neural networks II

For the second moment, it can be shown that

$$\tau^{-1}\mathbf{I} + \frac{1}{T}\sum_{t=1}^{T} \mathbf{f}^{\hat{\mathbf{w}}_t}(\mathbf{x}^*)^\top \mathbf{f}^{\hat{\mathbf{w}}_t}(\mathbf{x}^*) \xrightarrow{T\to\infty} \mathbb{E}_{q_\theta^*(\mathbf{y}^*|\mathbf{x}^*)}[(\mathbf{y}^*)^\top(\mathbf{y}^*)]$$

This means if we do T forward pass, we can get an estimation of the variance of approximate predictive distribution.

Applications of Bayesian Deep Learning I

We have linked stochastic regularisation techniques to approximate inference in BNN.