Bayesian Deep Learning

Narada Warakagoda

Forsvarets Forskningsinstitutt

ndw@ffi.no

October 19, 2020

Overview

Probability Review

Probability and Statistics Basics

- Normal (Gaussian) Distribution

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2} |\mathbf{\Sigma}|^{1/2}} \exp\left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \mathbf{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\} = \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$$

- Categorical Distribution

$$P(x) = \prod_{i=1}^{k} p_i^{[x=i]}$$

- Sampling

$$\boldsymbol{x} \sim p(\boldsymbol{x})$$

Probability and Statistics Basics

- Independent variables

$$p\left(\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_k\right) = \prod_{i=1}^{k} p\left(\boldsymbol{x}_i\right)$$

- Expectation

$$\mathbb{E}_{p(\boldsymbol{x})} f\left(\boldsymbol{x}\right) = \int f\left(\boldsymbol{x}\right) p\left(\boldsymbol{x}\right) dx$$

or for discrete variables

$$\mathbb{E}_{p(\boldsymbol{x})} f\left(\boldsymbol{x}\right) = \sum_{i=1}^{k} f\left(\boldsymbol{x}_i\right) P\left(\boldsymbol{x}_i\right)$$

Kullback Leibler Distance

$$KL\left(q\left(\boldsymbol{x}\right)||p\left(\boldsymbol{x}\right)\right) = \mathbb{E}_{q(\boldsymbol{x})} \log\left[\frac{q\left(\boldsymbol{x}\right)}{p\left(\boldsymbol{x}\right)}\right]$$
$$= \int \left[q\left(\boldsymbol{x}\right)\log q\left(\boldsymbol{x}\right) - q\left(\boldsymbol{x}\right)\log p\left(\boldsymbol{x}\right)\right] d\boldsymbol{x}$$

For the discrete case

$$KL\left(Q\left(\boldsymbol{x}\right)||P\left(\boldsymbol{x}\right)\right) = \sum_{i=1}^{k}\left[Q\left(\boldsymbol{x}_i\right)\log Q\left(\boldsymbol{x}_i\right) - Q\left(\boldsymbol{x}_i\right)\log P\left(\boldsymbol{x}_i\right)\right]$$

Bayesian Deep Learning

Bayesian Statistics

- Joint distribution

$$p(\boldsymbol{x}, \boldsymbol{y}) = p(\boldsymbol{x}|\boldsymbol{y})\, p(\boldsymbol{y})$$

- Marginalization

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}, \boldsymbol{y})\, d\boldsymbol{y}$$

$$P(\boldsymbol{x}) = \sum_{\boldsymbol{y}} P(\boldsymbol{x}, \boldsymbol{y})$$

- Conditional distribution

$$p(\boldsymbol{x}|\boldsymbol{y}) = \frac{p(\boldsymbol{x}, \boldsymbol{y})}{p(\boldsymbol{y})} = \frac{p(\boldsymbol{y}|\boldsymbol{x})\, p(\boldsymbol{x})}{\int p(\boldsymbol{y}|\boldsymbol{x})\, p(\boldsymbol{x})\, d\boldsymbol{x}}$$

Statistical view of Neural Networks

- Prediction

$$p\left(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w}\right)=\mathcal{N}\left(\boldsymbol{f}_{\boldsymbol{w}}\left(\boldsymbol{x}\right),\boldsymbol{\Sigma}\right)$$

- Classification

$$P\left(y|\boldsymbol{x},\boldsymbol{w}\right)=\prod_{i=1}^{k}\boldsymbol{f}_{\boldsymbol{w}}^{i}\left(\boldsymbol{x}\right)^{[y=i]}$$

Training Criteria

- Maximum Likelihood(ML)

$$\widehat{\boldsymbol{w}} = \arg\max_{\boldsymbol{w}} p\left(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{w}\right)$$

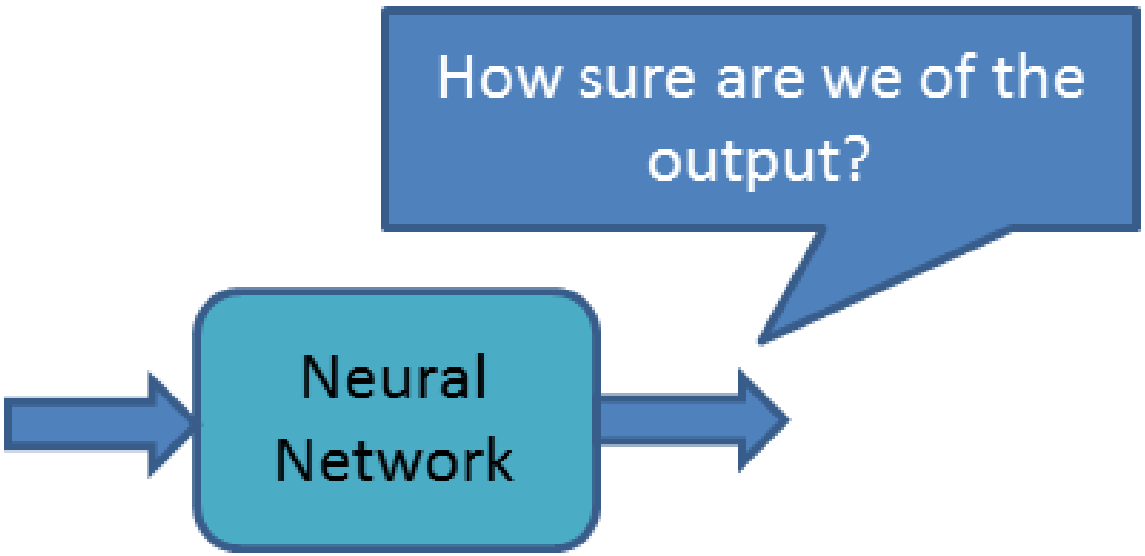
- Maximum A-Posteriori (MAP)

$$\widehat{\boldsymbol{w}} = \arg\max_{\boldsymbol{w}} p\left(\boldsymbol{Y}, \boldsymbol{w}|\boldsymbol{X}\right) = \arg\max_{\boldsymbol{w}} p\left(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{w}\right) p(\boldsymbol{w})$$

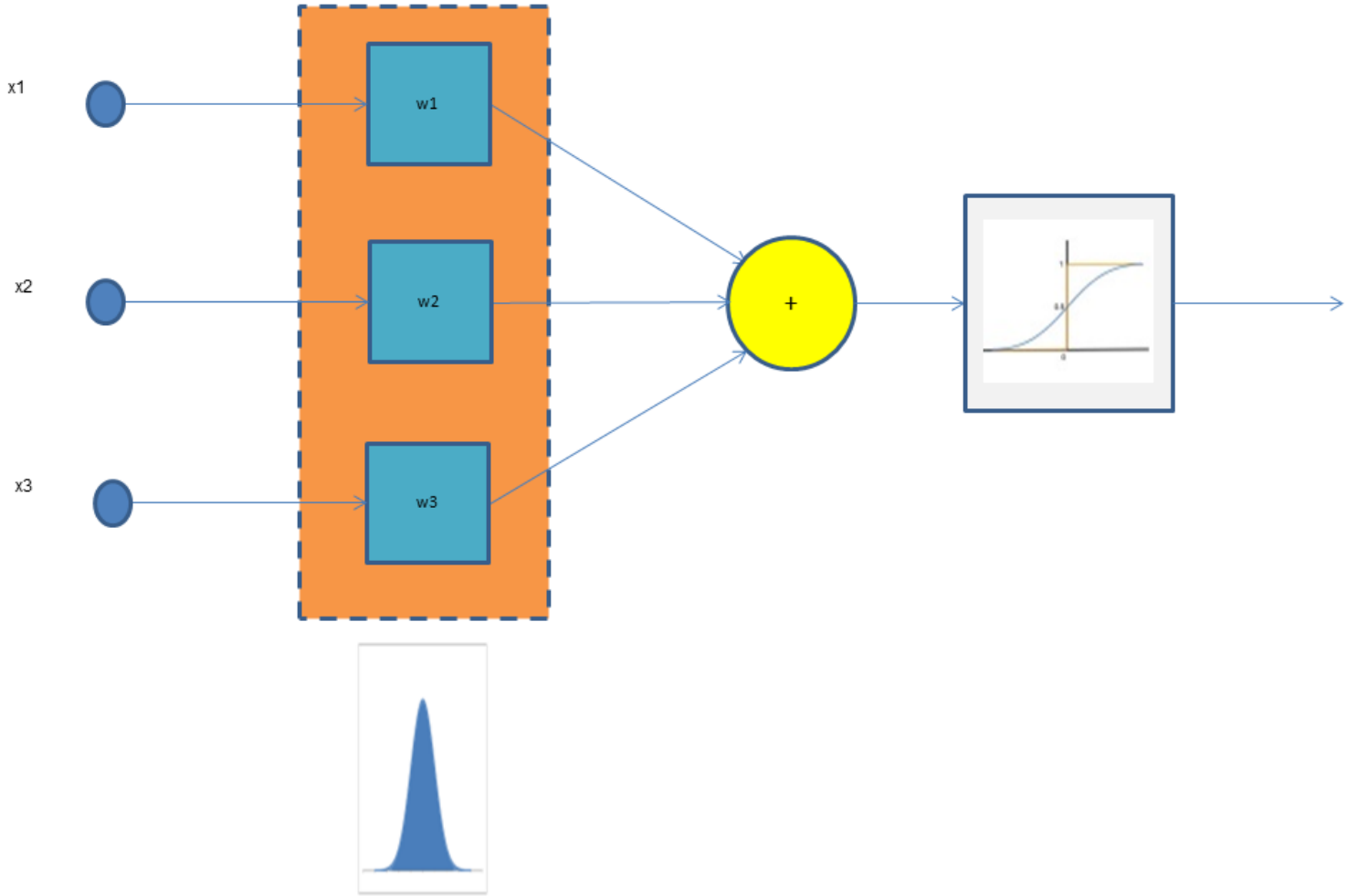
- Bayesian

$$p\left(\boldsymbol{w}|\boldsymbol{Y}, \boldsymbol{X}\right) = \frac{p\left(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{w}\right) p\left(\boldsymbol{w}\right)}{P\left(\boldsymbol{Y}|\boldsymbol{X}\right)} = \frac{p\left(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{w}\right) p\left(\boldsymbol{w}\right)}{\int P\left(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{w}\right) p\left(\boldsymbol{w}\right) d\boldsymbol{w}}$$

Motivation for Bayesian Approach



Motivation for Bayesian Approach



Types of Uncertainty

- Epistemic Uncertainty
  - Due to lack of data and modelling error.
  - Model parameter distributions can tackle this.
  - We consider this.
- Aleatoric Uncertainty
  - Due to poor quality data.
  - Need to model observation quality (noise).
  - We do not consider this.

Uncertainty with Bayesian Approach

- Not only prediction/classification, but their uncertainty can also be calculated
  - Since we have $p(\boldsymbol{w}|\boldsymbol{Y},\boldsymbol{X})$ we can sample $\boldsymbol{w}$ and use each sample as network parameters in calculating the prediction/classification $p(\widehat{y}|\widehat{x},\boldsymbol{w})$) (i.e.network output for a given input).
  - Prediction/classification is the mean of $p(\widehat{y}|\widehat{x},\boldsymbol{w})$

$$p_{out} = p(\widehat{y}|\widehat{x},\boldsymbol{Y},\boldsymbol{X}) = \int p(\widehat{y}|\widehat{x},\boldsymbol{w})\, p(\boldsymbol{w}|\boldsymbol{Y},\boldsymbol{X})\, d\boldsymbol{w}$$

  - Uncertainty of prediction/classification is the variance of $p(\widehat{y}|\widehat{x},\boldsymbol{w})$

$$\text{Var}(p(\widehat{y}|\widehat{x},\boldsymbol{w})) = \int \left[p(\widehat{y}|\widehat{x},\boldsymbol{w}) - p_{out}\right]^2 p(\boldsymbol{w}|\boldsymbol{Y},\boldsymbol{X})\, d\boldsymbol{w}$$

- Uncertainty is important in safety critical applications (eg: self-driving cars, medical diagnosis, military applications

Bayesian Approach vs ML and MAP

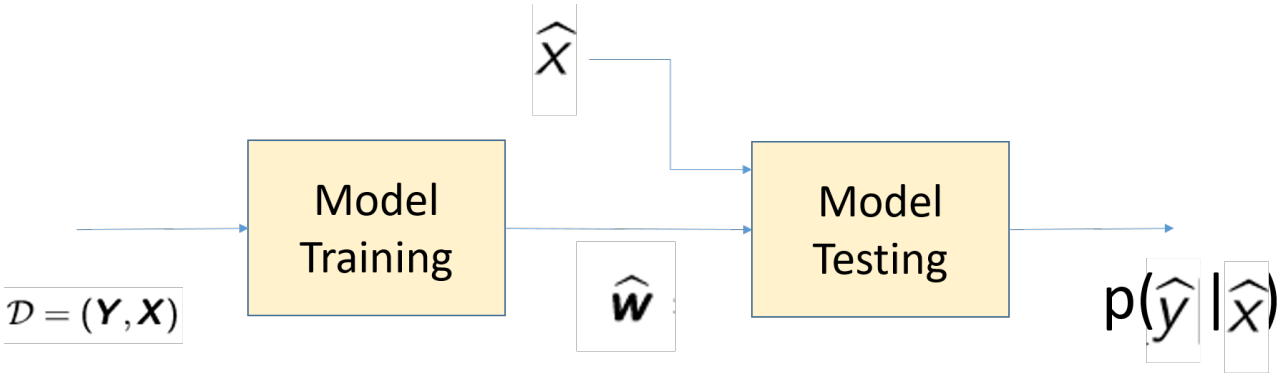


Figure: ML and MAP

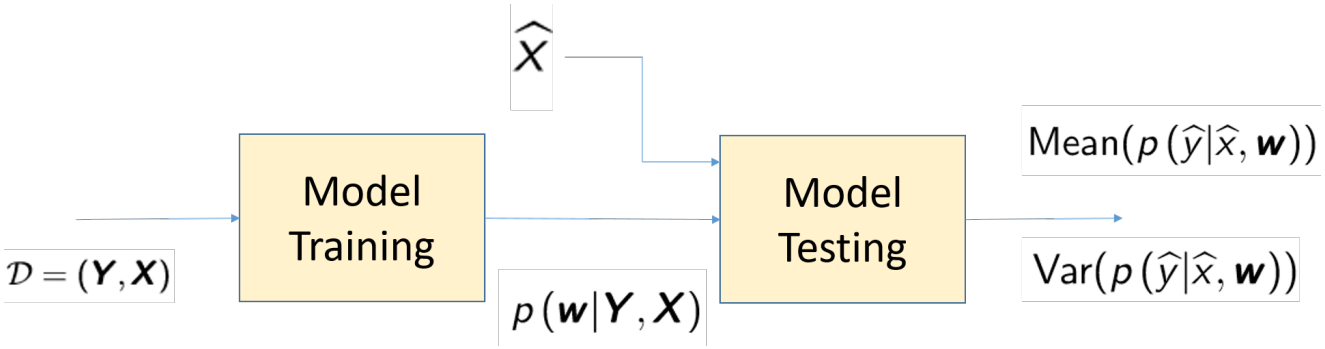


Figure: Bayesian approach

Other Advantages of Bayesian Approach

- Natural interpretation for regularization
- Model selection
- Input data selection (active learning)

Main Challenge of Bayesian Approach

- We calculate
  - For continuous case:

$$p(\boldsymbol{w}|\boldsymbol{Y},\boldsymbol{X}) = \frac{p(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{w})\,p(\boldsymbol{w})}{\int P(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{w})\,p(\boldsymbol{w})\,d\boldsymbol{w}}$$

  - For discrete case:

$$P(\boldsymbol{w}|\boldsymbol{Y},\boldsymbol{X}) = \frac{p(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{w})\,P(\boldsymbol{w})}{\sum_{\boldsymbol{w}} p(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{w})\,P(\boldsymbol{w})}$$

- Calculating denominator is often intractable
  - Eg: Consider a weight vector $\boldsymbol{w}$ of 100 elements, each can have two values. Then there are $2^{100} = 1.2 \times 10^{30}$ different weight vectors. Compare this with universe's age 13.7 billion years.
- We need approximations

Different Approaches

- Monte Carlo techniques (Eg: Markov Chain Monte Carlo -MCMC)
- Variational Inference
- Ensembles (eg: Dropout)

Advantages and Disadvantages of Different Approaches

- Markov Chain Monte Carlo - MCMC
  - Asymptotically exact
  - Computationally expensive
- Variational Inference
  - No guarantee of exactness
  - Possibility for faster computation

Monte Carlo Techniques

- We are interested in

$$p_{out} = \text{Mean}(p(\widehat{y}|\widehat{x}, \boldsymbol{w})) = p(\widehat{y}|\widehat{x}, \boldsymbol{Y}, \boldsymbol{X}) = \int p(\widehat{y}|\widehat{x}, \boldsymbol{w})\, p(\boldsymbol{w}|\boldsymbol{Y}, \boldsymbol{X})\, d\boldsymbol{w}$$

$$\text{Var}(p(\widehat{y}|\widehat{x}, \boldsymbol{w})) = \int [p(\widehat{y}|\widehat{x}, \boldsymbol{w}) - p_{out}]^2\, p(\boldsymbol{w}|\boldsymbol{Y}, \boldsymbol{X})\, d\boldsymbol{w}$$

- Both are integrals of the type

$$I = \int F(\boldsymbol{w})\, p(\boldsymbol{w}|\mathcal{D})\, d\boldsymbol{w}$$

where $\mathcal{D} = (\boldsymbol{Y}, \boldsymbol{X})$ is training data.

- Approximate the integral by sampling $\boldsymbol{w}_i$ from $p(\boldsymbol{w}|\mathcal{D})$

$$I \approx \frac{1}{L} \sum_{i=1}^{L} F(\boldsymbol{w}_i)\,.$$

Monte Carlo techniques

- Challenge: We don't have the posterior

$$p\left(\boldsymbol{w}|\mathcal{D}\right) = p\left(\boldsymbol{w}|\boldsymbol{Y},\boldsymbol{X}\right) = \frac{p\left(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{w}\right)p\left(\boldsymbol{w}\right)}{\int P\left(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{w}\right)p\left(\boldsymbol{w}\right)d\boldsymbol{w}}$$

- "Solution": Use importance sampling by sampling from a proposal distribution $q(\boldsymbol{w})$

$$I = \int F\left(\boldsymbol{w}\right)\frac{p\left(\boldsymbol{w}|\mathcal{D}\right)}{q\left(\boldsymbol{w}\right)}q\left(\boldsymbol{w}\right)d\boldsymbol{w} \approx \frac{1}{L}\sum_{i=}^{L} F\left(\boldsymbol{w}_i\right)\frac{p\left(\boldsymbol{w}_i|D\right)}{q\left(\boldsymbol{w}_i\right)}$$

- Problem: We still do not have $p\left(\boldsymbol{w}|\mathcal{D}\right)$

Monte Carlo Techniques

- Problem: We still do not have $p(\boldsymbol{w}|\mathcal{D})$
- Solution: use unnormalized posterior $\tilde{p}(\boldsymbol{w}|\mathcal{D}) = p(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{w})\, p(\boldsymbol{w})$ where normalization factor $Z = \int P(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{w})\, p(\boldsymbol{w})\, d\boldsymbol{w}$ such that

$$p(\boldsymbol{w}|\mathcal{D}) = \frac{\tilde{p}(\boldsymbol{w}|\mathcal{D})}{Z}$$

- Integral can be calculated with:

$$I \approx \frac{\sum_{i=1}^{L} F(\boldsymbol{w}_i)\, \tilde{p}(\boldsymbol{w}_i|D)/q(\boldsymbol{w}_i)}{\sum_{i=1}^{L} \tilde{p}(\boldsymbol{w}_i|D)/q(\boldsymbol{w}_i)}$$

Weakness of Importance Sampling

- Proposal distribution must be close to the non-zero areas of original distribution $p(\boldsymbol{w}|\mathcal{D})$.
- In neural networks, $p(\boldsymbol{w}|\mathcal{D})$ is typically small except for few narrow areas.
- Blind sampling from $q(\boldsymbol{w})$ has a high chance that they fall outside non-zero areas of $p(\boldsymbol{w}|\mathcal{D})$
- We must actively try to get samples that lie close to $p(\boldsymbol{w}|\mathcal{D})$
- Markov Chain Monte Carlo (MCMC) is such technique.

Metropolis Algorithm

- Metropolis algorithm is an example of MCMC
- Draw samples repeatedly from random walk $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon}$ is a small random vector, $\boldsymbol{\epsilon} \sim q(\boldsymbol{\epsilon})$ (eg: Gaussian noise)
- Drawn sample at $t = t$ is either accepted based on the ratio $\frac{\tilde{p}(\boldsymbol{w}_t|\mathcal{D})}{\tilde{p}(\boldsymbol{w}_{t-1}|\mathcal{D})}$
  - If $\tilde{p}(\boldsymbol{w}_t|\mathcal{D}) > \tilde{p}(\boldsymbol{w}_{t-1}|\mathcal{D})$ accept sample
  - If $\tilde{p}(\boldsymbol{w}_t|\mathcal{D}) < \tilde{p}(\boldsymbol{w}_{t-1}|\mathcal{D})$ accept sample with probability $\frac{\tilde{p}(\boldsymbol{w}_t|\mathcal{D})}{\tilde{p}(\boldsymbol{w}_{t-1}|\mathcal{D})}$
  - If sample accepted use it for calculating I
- Because $\frac{\tilde{p}(\boldsymbol{w}_t|\mathcal{D})}{\tilde{p}(\boldsymbol{w}_{t-1}|\mathcal{D})} = \frac{\frac{\tilde{p}(\boldsymbol{w}_t|\mathcal{D})}{p(\mathcal{D})}}{\frac{\tilde{p}(\boldsymbol{w}_{t-1}|\mathcal{D})}{p(\mathcal{D})}} = \frac{p(\boldsymbol{w}_t|\mathcal{D})}{p(\boldsymbol{w}_{t-1}|\mathcal{D})}$, sampling is valid for $p(\boldsymbol{w}|\mathcal{D})$ too.
- Since we sample $\boldsymbol{w}_i$ from $p(\boldsymbol{w}|\mathcal{D})$, approximate the integral with

$$I \approx \frac{1}{L} \sum_{i=1}^{L} F(\boldsymbol{w}_i).$$

Other Monte Carlo and Related Techniques

- Hybrid Monte Carlo (Hamiltonian Monte Carlo)
  - Similar to Metropolis algorithm
  - But uses gradient information rather than a random walk.
- Simulated Annealing

Variational Inference

- Goal: computation of posterior $p(\boldsymbol{w}|\mathcal{D})$, i.e. the parameters of the neural network $\boldsymbol{w}$ given data $\mathcal{D} = (\boldsymbol{Y}, \boldsymbol{X})$
- But this computation is often intractable
- Idea: find a distribution $q(\boldsymbol{w})$ from a family of distributions Q such that $q(\boldsymbol{w})$ can closely approximate $p(\boldsymbol{w}|\mathcal{D})$
- How to measure the distance between $q(\boldsymbol{w})$ and $p(\boldsymbol{w}|\mathcal{D})$?
  - Kullback-Leibler Distance $\mathrm{KL}\big(q(\boldsymbol{w})||p(\boldsymbol{w}|\mathcal{D})\big)$
- The problem can be formulated as

$$\hat{p}(\boldsymbol{w}|\mathcal{D}) = \arg\min_{q(\boldsymbol{w})} \mathrm{KL}\big(q(\boldsymbol{w})||p(\boldsymbol{w}|\mathcal{D})\big)$$

Minimizing KL Distance

- Using the definition of KL distance

$$\mathrm{KL}\big(q(\boldsymbol{w})||p(\boldsymbol{w}|\mathcal{D})\big) = \int q\left(\boldsymbol{w}\right) \ln \frac{q\left(\boldsymbol{w}\right)}{p\left(\boldsymbol{w}|\mathcal{D}\right)} d\boldsymbol{w}$$

- Cannot minimize this directly, because we do not know $p\left(\boldsymbol{w}|\mathcal{D}\right)$
- But we can manipulate it further, and transform it to another equivalent optimization problem involving a quantity known as Evidence Lower Bound (ELBO)

Evidence Lower Bound (ELBO)

$$
\begin{aligned}
\mathrm{KL}\big(q(\boldsymbol{w})||p(\boldsymbol{w}|\mathcal{D})\big) &= \int q(\boldsymbol{w}) \ln \frac{q(\boldsymbol{w})}{p(\boldsymbol{w}|\mathcal{D})} d\boldsymbol{w} \\
&= \int q(\boldsymbol{w}) \ln \frac{q(\boldsymbol{w})\, p(\mathcal{D})}{p(\boldsymbol{w}, \mathcal{D})} d\boldsymbol{w} \\
&= \int q(\boldsymbol{w}) \ln \frac{q(\boldsymbol{w})}{p(\boldsymbol{w}, \mathcal{D})} d\boldsymbol{w} + \int q(\boldsymbol{w}) \ln p(\mathcal{D}) d\boldsymbol{w} \\
&= \mathbb{E}_{q(\boldsymbol{w})} \ln \frac{q(\boldsymbol{w})}{p(\boldsymbol{w}, \mathcal{D})} + \ln p(\mathcal{D}) \int q(\boldsymbol{w})\, d\boldsymbol{w} \\
\ln p(\mathcal{D}) &= \boxed{\mathbb{E}_{q(\boldsymbol{w})} \ln \frac{p(\boldsymbol{w}, \mathcal{D})}{q(\boldsymbol{w})}} + \mathrm{KL}\big(q(\boldsymbol{w})||p(\boldsymbol{w}|\mathcal{D})\big)
\end{aligned}
$$

- Since $\ln p(\mathcal{D})$ is constant, minimizing $\mathrm{KL}\big(q(\boldsymbol{w})||p(\boldsymbol{w}|\mathcal{D})\big)$ is equivalent to maximizing ELBO

Another Look at ELBO

$$
\begin{aligned}
\text{ELBO} &= \mathbb{E}_{q(\boldsymbol{w})} \ln \frac{p(\boldsymbol{w}, \mathcal{D})}{q(\boldsymbol{w})} \\
&= \int q(\boldsymbol{w}) \ln p(\boldsymbol{w}, \mathcal{D}) d\boldsymbol{w} - \int q(\boldsymbol{w}) \ln q(\boldsymbol{w}) d\boldsymbol{w} \\
&= \int q(\boldsymbol{w}) \ln [p(\mathcal{D}|\boldsymbol{w}) p(\boldsymbol{w})] d\boldsymbol{w} - \int q(\boldsymbol{w}) \ln q(\boldsymbol{w}) d\boldsymbol{w} \\
&= \int q(\boldsymbol{w}) \ln p(\mathcal{D}|\boldsymbol{w}) d\boldsymbol{w} - \int q(\boldsymbol{w}) \ln \frac{q(\boldsymbol{w})}{p(\boldsymbol{w})} d\boldsymbol{w} \\
&= \mathbb{E}_{q(\boldsymbol{w})} \ln p(\mathcal{D}|\boldsymbol{w}) - \mathrm{KL}\big(q(\boldsymbol{w}) || p(\boldsymbol{w})\big)
\end{aligned}
$$

- We maximize ELBO with respect to $q(\boldsymbol{w})$
- First term $\mathbb{E}_{q(\boldsymbol{w})} \ln p(\mathcal{D}|\boldsymbol{w})$ is equivalent to maximizing $q(\boldsymbol{w})$'s ability explain training data
- Second term $\mathrm{KL}\big(q(\boldsymbol{w}) || p(\boldsymbol{w})\big)$ is equivalent to minimizing $q(\boldsymbol{w})$'s distance to $p(\boldsymbol{w})$

Outline of Procedure with ELBO

- Start with ELBO

$$\text{ELBO} = \mathcal{L} = \mathbb{E}_{q(\boldsymbol{w})} \ln \frac{p\left(\boldsymbol{w}, \mathcal{D}\right)}{q\left(\boldsymbol{w}\right)} = \mathbb{E}_{q(\boldsymbol{w})} \left[\ln p\left(\boldsymbol{w}, \mathcal{D}\right) - \ln q\left(\boldsymbol{w}\right) \right]$$

- Rewrite with parameter λ of $q\left(\boldsymbol{w}\right)$ and expand expectation

$$\mathcal{L}(\lambda) = \int \ln[p\left(\boldsymbol{w}, \mathcal{D}\right)] q\left(\boldsymbol{w}, \lambda\right) d\boldsymbol{w} - \int \ln[q\left(\boldsymbol{w}, \lambda\right)] q\left(\boldsymbol{w}, \lambda\right) d\boldsymbol{w}$$

- Maximize $\mathcal{L}(\lambda)$ with respect to λ

$$\lambda^{\star} = \arg\max_{\lambda} \mathcal{L}(\lambda)$$

- Use the optimized q witn respect to λ as posterior

$$q\left(\boldsymbol{w}, \lambda^{\star}\right) = p(\boldsymbol{w}, \mathcal{D})$$

How to Maximize ELBO

- Analytical methods are not practical for deep neural networks
- We resort to gradient methods with Monte Carlo sampling
- We discuss two methods:
  - Black box variational inference: Based on log derivative trick
  - Bayes by Backprop: Based on re-parameterization trick

Black Box Variational Inference

- Start with ELBO:

$$\mathcal{L}(\lambda) = \int \ln[p(\boldsymbol{w}, \mathcal{D})] q(\boldsymbol{w}, \lambda)\, d\boldsymbol{w} - \int \ln[q(\boldsymbol{w}, \lambda)] q(\boldsymbol{w}, \lambda)\, d\boldsymbol{w}$$

- Differentiate with respect to λ.

$$\begin{aligned}\nabla_\lambda \mathcal{L}(\lambda) = & \int \ln[p(\boldsymbol{w}, \mathcal{D})] \nabla_\lambda [q(\boldsymbol{w}, \lambda)] d\boldsymbol{w} \\ & - \int \ln[q(\boldsymbol{w}, \lambda)] \nabla_\lambda [q(\boldsymbol{w}, \lambda)] d\boldsymbol{w} \\ & - \int \nabla_\lambda \big[\ln[q(\boldsymbol{w}, \lambda)] \big] q(\boldsymbol{w}, \lambda)\, d\boldsymbol{w}\end{aligned}$$

- The last term is zero (Can you prove it?)

Black Box Variational Inference

- Now we have

$$\nabla_\lambda \mathcal{L}(\lambda) = \int \ln[p(\boldsymbol{w}, \mathcal{D})] \nabla_\lambda [q(\boldsymbol{w}, \lambda)] d\boldsymbol{w}$$
$$- \int \ln[q(\boldsymbol{w}, \lambda)] \nabla_\lambda [q(\boldsymbol{w}, \lambda)] d\boldsymbol{w}$$
$$= \int \Big[[\ln p(\boldsymbol{w}, \mathcal{D})] - \ln[q(\boldsymbol{w}, \lambda)] \Big] \nabla_\lambda [q(\boldsymbol{w}, \lambda)] dw$$

- We want to write this as an expectation with respect to q
- Use the log derivative trick

$$\nabla_\lambda [q(\boldsymbol{w}, \lambda)] = \nabla_\lambda [\ln q(\boldsymbol{w}, \lambda)] q(\boldsymbol{w}, \lambda)$$

Black Box Variational Inference

- Using log derivative trick, we get

$$\nabla_\lambda \mathcal{L}(\lambda) = \int \Big[\ln[p(\boldsymbol{w}, \mathcal{D})] - \ln q(\boldsymbol{w}, \lambda) \Big] \nabla_\lambda [\ln q(\boldsymbol{w}, \lambda)] q(\boldsymbol{w}, \lambda)\, d\boldsymbol{w}$$

- This is the same as Expectation with respect to q

$$\nabla_\lambda \mathcal{L}(\lambda) = \mathbb{E}_{q(\boldsymbol{w},\lambda)} \Big[\ln[p(\boldsymbol{w}, \mathcal{D})] - \ln q(\boldsymbol{w}, \lambda) \Big] \nabla_\lambda [\ln q(\boldsymbol{w}, \lambda)]$$

BBVI optimization procedure

- Assume a distribution $q(\boldsymbol{w}, \lambda)$ parameterized by λ.
- Draw S samples of $\boldsymbol{w}$ from the distribution using the current value of $\lambda = \lambda_t$
- Estimate the gradient of ELBO using the sample values:

$$\nabla_\lambda \hat{\mathcal{L}}(\lambda) = \frac{1}{S} \sum_{s=1}^{S} \Big[\ln[p(\boldsymbol{w}^s, \mathcal{D})] - \ln q(\boldsymbol{w}^s, \lambda) \Big] \nabla_\lambda [\ln q(\boldsymbol{w}^s, \lambda)]$$

- Update λ

$$\lambda_{t+1} = \lambda_t + \rho \nabla_\lambda \hat{\mathcal{L}}(\lambda)$$

- repeat from step 2

Bayes by Backprop

- Try to approximate ELBO directly by sampling from the $q(\boldsymbol{w}, \lambda)$

$$\text{ELBO} = \mathcal{L}(\lambda) = \mathbb{E}_{q(\boldsymbol{w},\lambda)}\big[\ln p\left(\boldsymbol{w}, \mathcal{D}\right) - \ln q\left(\boldsymbol{w}, \lambda\right)\big]$$

with

$$\hat{\mathcal{L}}(\lambda) = \frac{1}{S}\sum_{s=1}^{S}\big[\ln p\left(\boldsymbol{w}^s, \mathcal{D}\right) - \ln q\left(\boldsymbol{w}^s, \lambda\right)\big]$$

- But we need $\nabla_\lambda \hat{\mathcal{L}}(\lambda)$ and we can not differentiate $\hat{\mathcal{L}}(\lambda)$ because it is not a smooth function of λ
- Use the re-parameterization trick

$$\boldsymbol{w}^s = \boldsymbol{w}(\lambda, \boldsymbol{\epsilon}^s)$$

where $\boldsymbol{\epsilon}^s$ is drawn from for example a standard Gaussian distribution.

Bayes by BackProp (BbB)

- The estimated ELBO now

$$\hat{\mathcal{L}}(\lambda) = \frac{1}{S}\sum_{s=1}^{S}\left[\ln p\left(\boldsymbol{w}(\lambda, \boldsymbol{\epsilon}^s), \mathcal{D}\right) - \ln q\left(\boldsymbol{w}(\lambda, \boldsymbol{\epsilon}^s), \lambda\right)\right]$$

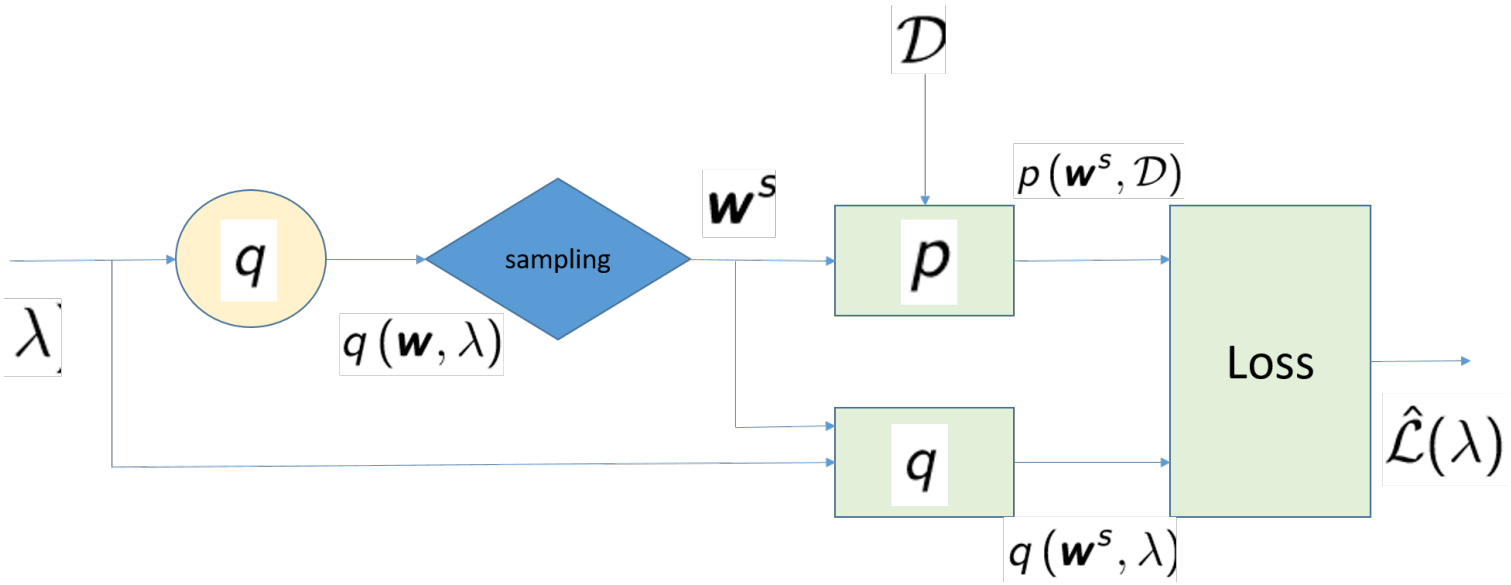
- Now this is a smooth function of λ and can differentiate

$$\nabla_\lambda \hat{\mathcal{L}}(\lambda) = \frac{1}{S}\sum_{s=1}^{S}\left[\frac{\partial \hat{\mathcal{L}}_s}{\partial \boldsymbol{w}}\frac{\partial \boldsymbol{w}}{\partial \lambda} + \frac{\partial \hat{\mathcal{L}}_s}{\partial \lambda}\right]$$

  where $\hat{\mathcal{L}}_s = \ln p\left(\boldsymbol{w}(\lambda, \boldsymbol{\epsilon}^s), \mathcal{D}\right) - \ln q\left(\boldsymbol{w}(\lambda, \boldsymbol{\epsilon}^s), \lambda\right)$

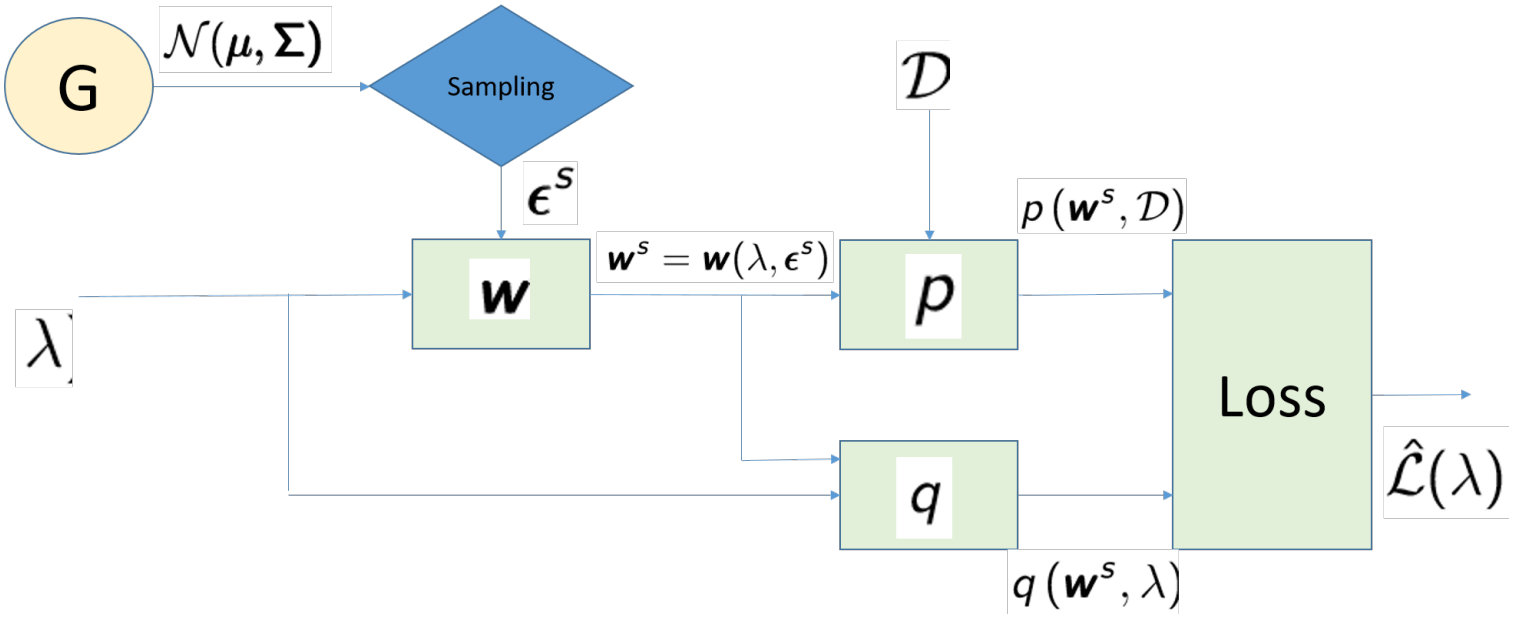
- Once the gradients are known, optimum $\lambda^\star$ and hence $q(\boldsymbol{w}, \lambda^\star)$ can be found by gradient descent.

Bayes by Backprop - Schematic 1



$$\hat{\mathcal{L}}(\lambda) = \frac{1}{S} \sum_{s=1}^{S} \left[\ln p(\boldsymbol{w}^s, \mathcal{D}) - \ln q(\boldsymbol{w}^s, \lambda) \right]$$

Bayes by Backprop - Schematic 2



$$\hat{\mathcal{L}}(\lambda) = \frac{1}{S} \sum_{s=1}^{S} \left[\ln p\left(\boldsymbol{w}(\lambda, \boldsymbol{\epsilon}^s), \mathcal{D}\right) - \ln q\left(\boldsymbol{w}(\lambda, \boldsymbol{\epsilon}^s), \lambda\right) \right]$$

Performance of BBVI and BbB

- Both methods estimate approximate gradients by sampling
- High variance of the estimated gradients is a problem
- In practice, these algorithms need modifications to tackle high variance
- BbB tends to have a lower variance estimates than BBVI

Bayesian Deep Learning through Ensembles

- Direct ensembles
- Indirect ensembles- Dropout

Bayesian Deep Learning with Direct Ensembles

- Train a set of models (say S models) with the same data set, but with different sets of initial values.
- Feed each network S with the test data and collect the outputs $f(s)$, $s = 1, 2, \cdots, S$
- Output variance $= \frac{1}{S}\sum_s (f(s) - \bar{f}(s))^2$ where $\bar{f}(s) = \frac{1}{S}\sum_s f(s)$

Bayesian Deep Learning with Dropout

- Stochastic gradient descent and Dropout can be given Bayesian interpretations
- Dropout procedure in testing can be used for estimating the uncertainty of model outputs (Monte Carlo Dropout).
  - Enable dropout and feed the network S times with test data and collect the outputs $f(s)$, $s = 1, 2, \cdots, S$
  - Output variance $= \frac{1}{S}\sum_s (f(s) - \bar{f}(s))^2$ where $\bar{f}(s) = \frac{1}{S}\sum_s f(s)$