Toward Bayesian Deep Learning

MI2RL 장령우

Yarin Gal

- Godfather of modern Bayesian Deep Learning
- Ph.D thesis : “Uncertainty of Deep Learning”(http://mlg.eng.cam.ac.uk/yarin/thesis/thesis.pdf)

Uncertainty in Deep Learning

- Bayes' Rule

$$f(x|y) = \frac{f(y|x)f(x)}{f(y)} \propto \mathcal{L}(y|x)f(x)$$

- f(x|y) : posterior probability
- f(x) : prior probability of X
- f(y) : prior probability of Y (normalizing constant)
- f(y|x) : conditional probability given X
- L(y|x) : likelihood

Uncertainty in Deep Learning

- Types of uncertainty
  - Noisy data : Observations can be noisy d/t measurement imprecision -> Aleatoric uncertainty
  - Uncertainty in model parameters : Large number of models can be used to explain train data. Which model to use? -> Epistemic uncertainty 1
  - Structural uncertainty : What model structure to use? -> Epistemic uncertainty 2
- Predictive uncertainty = Aleatoric uncertainty + Epistemic uncertainty

(confidence we have in prediction)

- Epistemic : Came from “episteme”, Greek for “knowledge” -> “reducible unc-”
- Aleatoric : Came from “aleator”, Latin for “dice player” -> “irreducible unc-

Uncertainty in Deep Learning

- Deep learning model does not offer uncertainties!
- Does softmax output can be interpreted as model confidence? (classification)
    - A model can be wrong although having high softmax

Uncertainty in Deep Learning

- Given a dataset X,Y, we look for the *posterior* distribution over the space of parameters by invoking Bayes' theorem:

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

- After training, we can predict new data as follows (inference):

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

- Model evidence:

$$p(Y|X) = \int p(Y|X,\omega)p(\omega)d\omega$$

Uncertainty in Deep Learning

- True posterior $p(\omega|X, Y)$ cannot be computed analytically.
- Instead, we define a *variational* distribution $q_\theta(\omega)$ which is easy to evaluate.
- Thus, we minimize Kullback-Leibler (KL) divergence w.r.t θ :

$$KL(q_\theta(\omega)||p(\omega|X,Y)) = \int q_\theta(\omega) \log \frac{q_\theta(\omega)}{p(\omega|X,Y)} d\omega$$

- Minimising KL divergence allows us for :

$$\begin{aligned} p(y^*|x^*, X, Y) &= \int p(y^*|x^*, \omega) p(\omega|X, Y) d\omega \\ &\approx \int p(y^*|x^*, \omega) q_\theta^*(\omega) d\omega =: q_\theta^*(y^*|x^*) \end{aligned}$$

Uncertainty in Deep Learning

- KL divergence minimising == Maximising *evidence lower bound (ELBO)* w.r.t variational parameters defining $q_\theta(\omega)$

$$\mathcal{L}_{VI}(\theta) := \int q_\theta(\omega) \log p(Y|X,\omega)$$
$$\leq \log p(Y|X)$$

where log p(Y|X) is “log evidence”, and VI is “Variational Inference”

Uncertainty in Deep Learning

- As

$$KL\Big(q_\theta(\omega)||p(\omega|X,Y)\Big) \propto -\int q_\theta(\omega)\log p(Y|X,\omega)d\omega + KL(q_\theta(\omega)||p(\omega))$$
$$= -\sum_{i=1}^{N}\int q_\theta(\omega)\log p(y_i|f^\omega(x_i))d\omega + KL(q_\theta(\omega)||p(\omega))$$

- We set

$$\mathcal{L}_{VI}(\theta) := KL\Big(q_\theta(\omega)||p(\omega|X,Y)\Big)$$

Uncertainty in Deep Learning

- Therefore,

$$\mathcal{L}_{VI}(\theta) := -\sum_{i=1}^{N} \int q_\theta(\omega) \log p(y_i | f^\omega(x_i)) d\omega + KL(q_\theta(\omega) || p(\omega))$$

- This has two problems:
    - First term on RHS is not tractable with more than single hidden layer.
    - This objective requires computation over the whole dataset, which is computationally expensive.
- Therefore, mini-batch optimisation (data sub-sampling)

$$\widehat{\mathcal{L}}_{VI}(\theta) := -\frac{N}{M} \sum_{i \in S} \int q_\theta(\omega) \log p(y_i | f^\omega(x_i)) d\omega + KL(q_\theta(\omega) || p(\omega))$$

where S is random index set with size M

Uncertainty in Deep Learning

- Data sub-sampling approximation forms an unbiased estimator, meaning

$$\mathbb{E}_S[\widehat{\mathcal{L}}_{VI}(\theta)] = \mathcal{L}_{VI}(\theta)$$

- Monte Carlo is often used for variational inference. Yet there is gradient descent method:

Algorithm 1 Minimise divergence between $q_\theta(\boldsymbol{\omega})$ and $p(\boldsymbol{\omega}|X, Y)$

Given dataset $\mathbf{X}, \mathbf{Y}$,
Define learning rate schedule η,
Initialise parameters θ randomly.
repeat
  Sample M random variables $\widehat{\boldsymbol{\epsilon}}_i \sim p(\boldsymbol{\epsilon})$, S a random subset of $\{1, .., N\}$ of size M.
  Calculate stochastic derivative estimator w.r.t. θ:

$$\widehat{\Delta\theta} \leftarrow -\frac{N}{M}\sum_{i \in S}\frac{\partial}{\partial\theta}\log p(\mathbf{y}_i|\mathbf{f}^{g(\theta,\widehat{\boldsymbol{\epsilon}}_i)}(\mathbf{x}_i)) + \frac{\partial}{\partial\theta}\mathrm{KL}(q_\theta(\boldsymbol{\omega})||p(\boldsymbol{\omega})).$$

  Update θ:

$$\theta \leftarrow \theta + \eta\widehat{\Delta\theta}.$$

until θ has converged.

Uncertainty in Deep Learning

- Stochastic Regulation Technique (SRT) : Dropout !
- $\hat{\epsilon}_i$: vector with dimension Q (input dimensionality) whose elements are random probabilities. (Dropout rate)
- $\hat{x} = x \odot \hat{\epsilon}_i$ where x is input vector.
- $\widehat{h} = \sigma(\widehat{x}M_1 + b) \odot \widehat{\epsilon}_2$
- Output : $\widehat{y} = \widehat{h}M_2$

Uncertainty in Deep Learning

- Therefore,

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

- Which implies,

$$\frac{\partial}{\partial\theta}\widehat{\mathcal{L}}_{\text{dropout}}(\theta) = \frac{1}{N_\tau}\frac{\partial}{\partial\theta}\widehat{\mathcal{L}}_{\text{MC}}(\theta)$$

- Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning(https://arxiv.org/pdf/1506.02142.pdf)

Uncertainty in Deep Learning

- Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning(https://arxiv.org/pdf/1506.02142.pdf)
    - Dropout NN is equivalent as probabilistic Gaussian Process
    - Uncertainty of dropout NN model is expressed as :

$$\log p(y^*|x^*, X, Y) \approx \log\left(\sum_{i=1}^{T} \exp(-\frac{1}{2}\tau||y - \hat{y}_t||^2)\right) - \log T - \frac{1}{2}\log 2\pi - \frac{1}{2}\log \tau^{-1}$$

- Gaussian process : A stochastic process that is composed with normal distributions: $X_{(t_1, t_2, \cdots, t_n)} = (X_{t_1}, X_{t_2}, \cdots, X_{t_n})$

What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?

- Epistemic Uncertainty in Bayesian Deep Learning :

$$p(y = c|x, X, Y) \approx \frac{1}{T}\sum_{t=1}^{T} \mathrm{Softmax}(f^{\widehat{W}_t}(x))$$

where uncertainty of this probability vector p is expressed as entropy H:

$$H(p) = -\sum_{c=1}^{C} p_c \log p_c$$

- Aleatoric Uncertainty :

$$\mathcal{L}_{NN}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{2\sigma(x_i)^2}||y_i - f(x_i)||^2 + \frac{1}{2}\log \sigma(x_i)^2$$

What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?

- Combining aleatoric and epistemic uncertainty in one model :

$$[\hat{y}, \hat{\sigma}^2] = f^{\widehat{W}}(x)$$

- To summarize, with $\hat{y}_t, \hat{\sigma}_t^2 = f^{\widehat{W}_t}(x)$, with T models and

$$\mathrm{Var}(y) \approx \frac{1}{T}\sum_{i=1}^{T} \hat{y}_t^2 - \left(\frac{1}{T}\sum_{t=1}^{T} \hat{y}_t\right)^2 + \frac{1}{T}\sum_{t=1}^{T} \hat{\sigma}_t^2$$

- Objective : To minimise

$$\mathcal{L}_{BNN}(\theta) = \frac{1}{D}\sum_{i} \frac{1}{2}\hat{\sigma}_i^{-2}||y_i - \hat{y}_i||^2 + \frac{1}{2}\log \hat{\sigma}_i^2$$