Weekly Study Report

Wang Ma

2024-09-06

Electrical, Computer, and Systems Engineering Department
Rensselaer Polytechnic Institute

Table of contents

Bayesian Deep Learning

What is a Bayesian Neural Network (BNN)

Definition:

A Bayesian neural network is a neural network that uses (approximate) Bayesian Inference for **uncertainty estimation**, i.e., we can treat the NN parameters as random variables and infer them using (approximate) Bayesian posterior inference.

Traditional DL Approach:

$$\theta^* = \arg\max_\theta \mathbb{E}_{(x,y)\sim[}[\log p(y|x,\theta)]$$

- Point Estimation
- May be overconfident in the prediction

  (outputs of softmax layer tend to close to 0 or 1). Can not imply the model's confidence or uncertainty in the prediction.

Bayes' Rule

In BNN, the network parameters θ are treated as random variables, and we perform Bayesian Inference on it. In details, we have the following formula which is called *Bayes' Rule*:

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)}, \quad p(D|\theta) = \Pi_{n=1}^{N} p(y_n|x_n, \theta).$$

- **Prior Distribution**: $p(\theta)$, our prior knowledge of NN parameters θ
- **Likelihood**: $p(D|\theta)$, how well the parameters explain/support observed data
- **Posterior**: $p(\theta|D)$, our belief/knowledge of θ after data observation
- **Evidence**: $p(D)$, to normalize the posterior

Bayesian Prediction

Bayesian Predictive distribution:

$$p(y^*|x^*, D) = \int p(y^*|x^*, \theta)p(\theta|D)d\theta.$$

Unfortunately we don't know how to directly compute $p(\theta|D)$ nor $p(y^*|x^*, D)$. Most of the existing approaches solve the problem in the following 3 steps:

- Design an approximate posterior $q(\theta) \in Q$ which is easy to compute and sample from;
- Fit $q(\theta) \approx p(\theta|D)$ (Variational Inference, ELBO)
- Approximate predictive inference with Monte Carlo:

$$p(y^*|x^*, D) \approx \int p(y^*|x^*, \theta)q(\theta)d\theta \approx \frac{1}{K}\sum_{k=1}^{K} p(y^*|x^*, \theta_k), \quad \theta_k \sim q(\theta).$$

Variational Inference to Approximate Posterior

Approximation: $q^*(\theta) = \arg\min_{q \in Q} KL[q(\theta) \parallel p(\theta|D)]$,

where

$$KL[q(\theta) \parallel p(\theta|D)] = \mathbb{E}_q[\log q(\theta) - \log p(\theta|D)].$$

Further derivation:

$$\log p(\theta|D) = \log \frac{p(D|\theta)p(\theta)}{p(D)} = \log p(D|\theta) + \log p(\theta) - \log p(D),$$

which means (notice that $\log p(D)$ is a constant w.r.t. q and θ):

$$\begin{aligned} KL[q(\theta)||p(\theta|D)] &= \mathbb{E}_q[\log q(\theta) - \log p(D|\theta) - \log p(\theta)] + \log p(D) \\ &= \log p(D) - (\mathbb{E}_q[\log p(D|\theta)] - KL[q(\theta)||p(\theta)]) \\ &:= \log p(D) - ELBO(q, D). \end{aligned}$$

In other words, the below optimisation problems are equivalent:

$$\min_{q \in Q} KL[q(\theta)||p(\theta|D)] \quad \Leftrightarrow \quad \max_{q \in Q} ELBO(q, D),$$

$$ELBO(q, D) = \mathbb{E}_q[\log p(D|\theta)] - KL[q(\theta)||p(\theta)].$$

Think about ELBO:

Evidence Lower Bound (ELBO):

$$ELBO(q, D) = \mathbb{E}_q[\log p(D|\theta)] - KL[q(\theta) \parallel p(\theta)]$$

- **Data fitting**: $\mathbb{E}_q[\log p(D|\theta)]$ measures on average how good neural networks with parameters sampled from q fit the training data.
- **Complexity regularization**: $KL[q(\theta) \parallel p(\theta)]$ describes the amount of changes of q from the prior p. In BNN literature the prior p on weights are often set to be less informative (e.g., Gaussian with zero mean and large variance), in such case the KL term can also be viewed as regularizing the complexity of q.

Tempered ELBO:

$$ELBO_\beta(q, D) = \mathbb{E}_q[\log p(D|\theta)] - \beta KL[q(\theta) \parallel p(\theta)]$$

Goal: to balance between the data fitting quality and the complexity .

Other ways to approximate posteror

Last-layer BNN

$$ELBO_\beta(q, D) = \mathbb{E}_q[\log p(D|\theta)] - \beta KL[q(\theta^L) \parallel p(\theta^L)]$$

Monte Carlo dropout

$$ELBO_\beta(q, D) = \mathbb{E}_q[\log p(D|\theta)] - (1 - \pi) l_2(\psi)$$

Laplace approximation

MCMC: Mrakov Chain Monte Carlo

Ensemble NNs

...

Bayesian Optimization

https://www.overleaf.com/1978916912xdpyrmhjvmsh#e83875

Uncertainty Measures

- **Epistemic uncertainty:** also named model uncertainty, this is the uncertainty due to **lack of knowledge**, and thus can be reduced by **collecting more data**. For example, by flipping a coin multiple times, we become more and more certain about whether the coin is fair or bent;
- **Aleatoric uncertainty:** also named data uncertainty, this is the uncertainty regarding **the stochasticity of individual experimental outcome**, which is **non-reducible**. For example, even if we are 100% sure about that the coin is fair, we are still unsure about whether the next coin flip result will be head or tail.

These two types of uncertainty, when summed, returns the total uncertainty, i.e.,

$$\text{total uncertainty} = \text{epistemic uncertainty} + \text{aleatoric uncertainty}.$$

Quantify Uncertainty

Shannon Entroy: $\mathbb{H}[p] = -\sum_{c=1}^{C} p_c \log p_c$.

Total Uncertainty:

$$\mathbb{H}[y^*|x^*, D] = \mathbb{H}[p(y^*|x^*, D)]$$
$$= -\sum p(y^* = c|x^*, D) \log p(y^* = c|x^*, D)$$

Aleatoric Uncertainty:

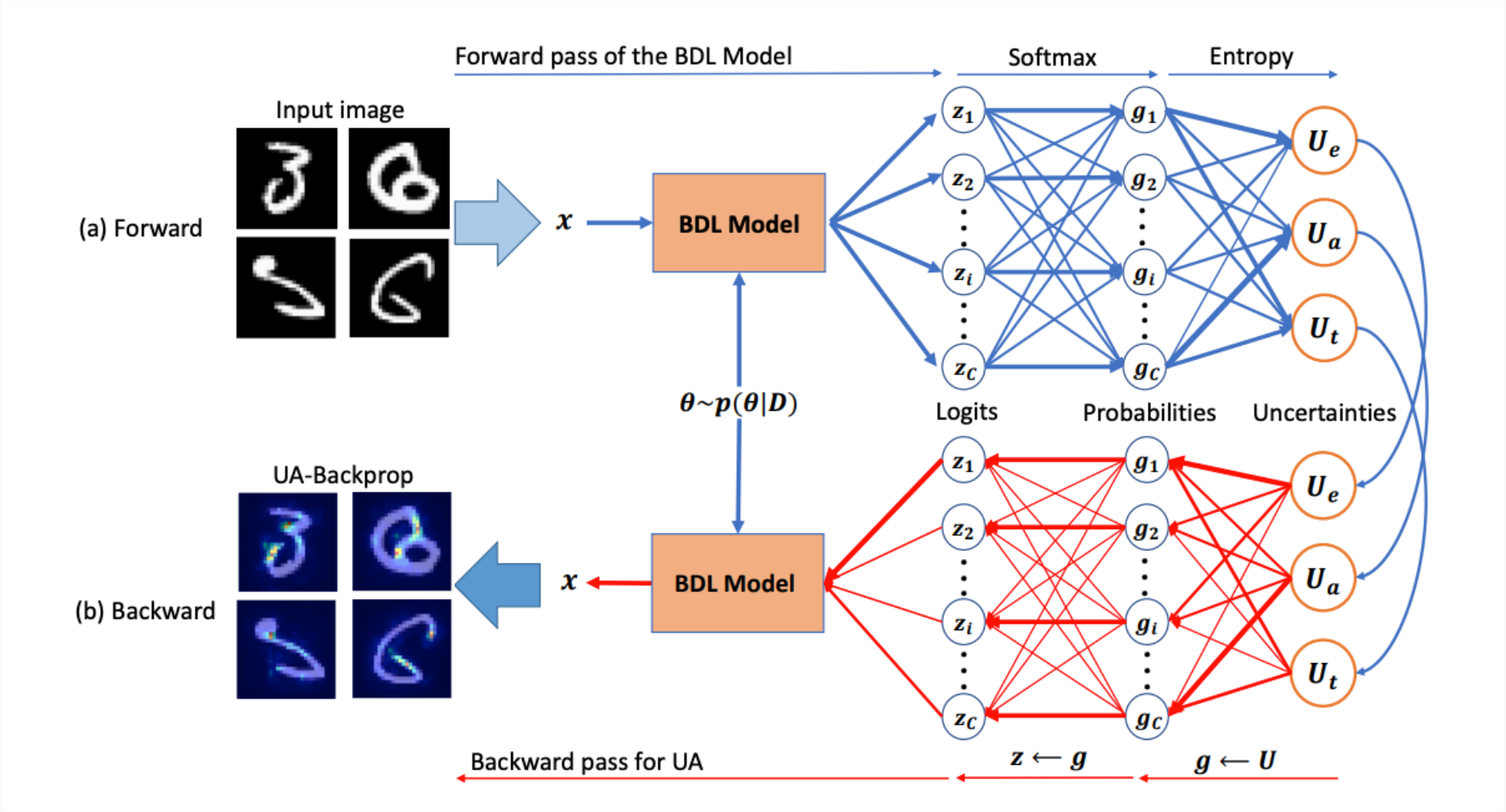
$$\mathbb{E}_{p(\theta|D)}\mathbb{H}[y^*|x^*, \theta] = \mathbb{E}_{p(\theta|D)}\mathbb{H}[p(y^*|x^*, \theta)]$$
$$= \mathbb{E}_{\mathbb{E}_{p(\theta|D)}}[-\sum p(y^* = c|x^*, D) \log p(y^* = c|x^*, D)]$$

Epistemic Uncertainty

$$\mathbb{I}[y^*; \theta \mid x^*, D] = \mathbb{H}[y^*|x^*, D] - \mathbb{E}_{p(\theta|D)}\mathbb{H}[y^*|x^*, \theta]$$
$$= \mathbb{E}_{p(y^*|x^*, D)}[KL[p(\theta|y^*, x^*, D) \| p(\theta|D)]]$$

Paper Reading: Gradient-based Uncertainty Attribution

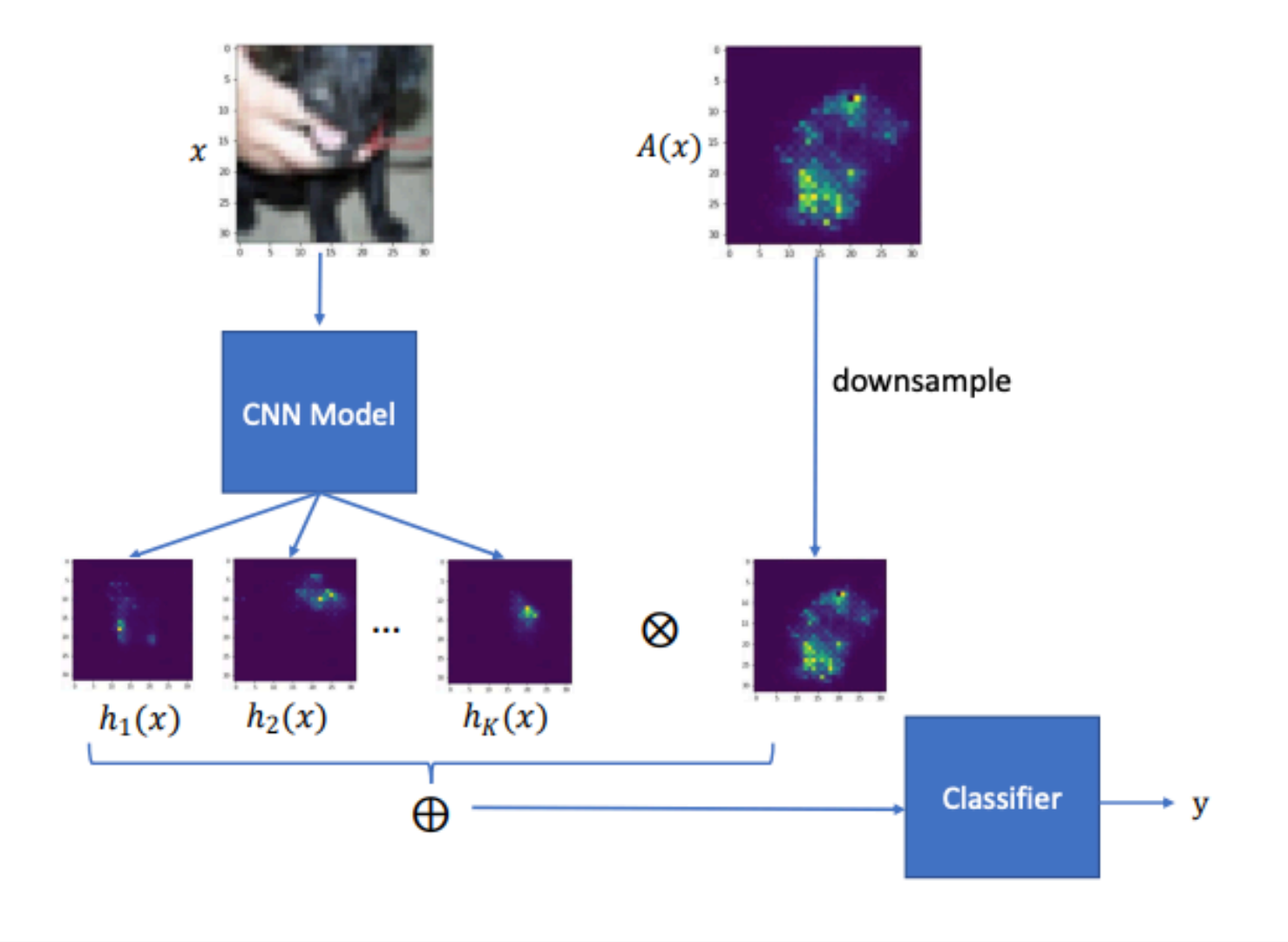
Gradient-based UA to Generate the Attribution Map



Uncertainty Mitigation via Attribution Map

Attribution Map: $M(x)$

Attention Weights: $A(x) = (1 - M(x)) \odot M(x)$



Paper Reading: Epistemic UQ for Pre-trained NNs

Proposition 3.1

$$\theta^* + \sigma\varepsilon \sim N(\theta^*, \sigma^2 I), \quad \text{where} \quad \varepsilon \sim N(0, I)$$

$$\text{and} \quad \sup_\theta |p(\theta|D) - N(\theta; \theta^*, \sigma^2 I)| \to 0$$

Proposition 3.2

For finite D, we have an upper bound for the distance between $p(\theta|D)$ and $N(\theta; \theta^*, \sigma^2 I)$, say, $D_{TV}[p(\theta|D) \| N(\theta; \theta^*, \sigma^2 I)]$.

Proposition 3.3

Equivalence between little perturbations on parameters and on inputs:

$$f(x, \theta + \Delta\theta) = f(x + \Delta x, \theta),$$

which implies

$$\begin{aligned} U_e(x) &= \mathbb{E}_{\Delta\theta}[KL(p(y|x, \theta^* + \Delta\theta) \| p(y|x, \theta^*))] \\ &= \mathbb{E}_{\Delta x}[KL(p(y|x + \Delta x, \theta^*) \| p(y|x, \theta^*))]. \end{aligned}$$

Proposition 3.4

$$\frac{\partial f(x,\theta^*)}{\partial \theta^*} = 0 = \frac{\partial f(x+\Delta x,\theta^*)}{\partial \theta^*} = 0$$

Proposition 3.5

Proposition 3.5. *The epistemic uncertainty derived by the expected gradient norm can serve as an upper bound compared to the uncertainty produced by perturbation-based methods when the perturbations are small.*

$$\begin{aligned}
&\mathbb{E}_{p(\Delta\theta)}\left[\mathbf{KL}(p(y|x,\theta^*)||p(y|x,\theta^*+\Delta\theta))\right] \\
&\leq \sum_{c=1}^{C} p(y=c|x,\theta^*) \left\| \frac{\partial \log p(y=c|x,\theta^*)}{\partial \theta^*} \right\| \mathbb{E}_{p(\Delta\theta)}[||\Delta\theta||] \\
&\propto \mathbb{E}_{y\sim p(y|x,\theta^*)} \left[\left\| \frac{\partial \log p(y|x,\theta^*)}{\partial \theta^*} \right\| \right] \text{(ExGrad [11])}
\end{aligned} \tag{10}$$

where $\Delta\theta \to 0$ *and* $\mathbb{E}_{p(\Delta\theta)}[||\Delta\theta||]$ *is independent of* x.

Proposed Method

a. Class-specific Gradient Wighting (from 3.5)

$$U_{\text{REGrad}}(x) = \sum_{c=1}^{C} \sqrt{p(y=c|x,\theta^*)} \left\| \frac{\partial \log p(y=c|x,\theta^*)}{\partial \theta^*} \right\|_2^2$$

b. Layer-selective Gradients

$$\left\| \frac{\partial \log p(y|x,\theta^*)}{\partial \theta^*} \right\| \xrightarrow[\text{selective}]{\text{layer}} \sum_{\theta_l^* \in \theta^*} a_l \left\| \frac{\partial \log p(y|x,\theta^*)}{\partial \theta_l^*} \right\|$$

c. Gradient Perturbation Integration (from 3.4)

$$\left\| \frac{\partial \log p(y|x_0,\theta^*)}{\partial \theta^*} \right\| \xrightarrow[\text{smoothed}]{\text{perturb}} \left\| \frac{1}{N+1} \sum_{i=0}^{N} \frac{\partial \log p(y|x_i,\theta^*)}{\partial \theta^*} \right\|$$

Plan for Next Week

Reading Plans

1. Naiyu's paper on "Quantifying Uncertainty in Causal Graphs"
2. Hanjing's paper on "Diversity-enhanced Probabilistic Ensemble For Uncertainty Estimation"
3. Hanjing's paper on "Semantic Attribution For Explainable Uncertainty"
4. The survey paper "A survey of uncertainty in deep neural networks"

Goal: Carefully read the first three papers, and then expand on the remaining ones and some other works

Ending

Thank you!

Quesitions are welcome!