β-BNN: A Rate-Distortion Perspective on Bayesian Neural Networks

Shell X. Hu[1], Pablo G. Moreno[2], Neil D. Lawrence[2] and Andreas Damianou[2]

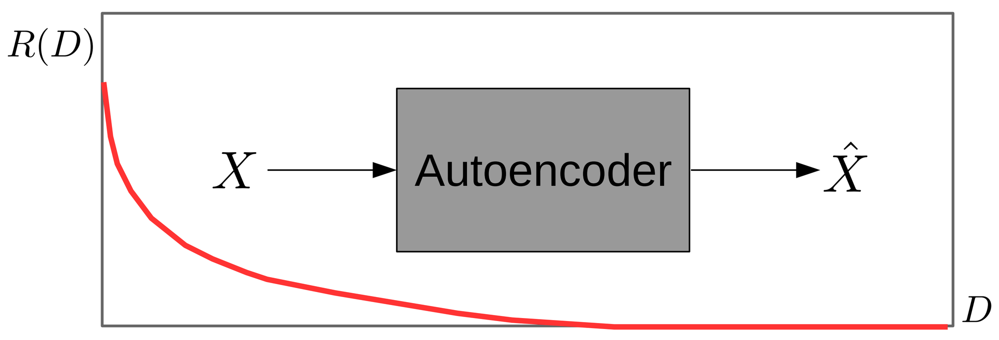
[1]École des Ponts ParisTech, Champs-sur-Marne, France
[2]Amazon, Cambridge, United Kingdom

Main idea

We propose an alternative training framework for Bayesian neural networks (BNNs), which is motivated by viewing the latent variable model for supervised learning as an autoencoder for data transmission. Then, a natural objective can be invoked from the rate-distortion theory leading to an iterative update on the "prior" and the "posterior".

Background: lossy compression

Goal: determine the minimal number of bits, denoted by R, to encode a signal X, such that the distortion of X yielded by the autoencoding does not exceed D.



Lossless compression $\Rightarrow$ BNN

This connection with lossless compression was established by [1] using the minimum description length (MDL) and the bits back argument for noisy weights:

$$\min_q \mathsf{KL}(q \| \text{prior}) + \mathbb{E}_q[\text{data misfit}],$$

which is considered as *vanilla BNN* when the *variational posterior* q is specified as diagonal Gaussian due to [2].

Model uncertainty: Bayesian vs. "Frequentist"

Bayesians describe data $S = \{(x_i, y_i)\}_{i=1}^n$ through generative decomposition of the latent variable model

$$p(S, w) = p(w)p(S|w) = p(w)\prod_i p(y_i|x_i, w_y)p(x_i|w_x).$$

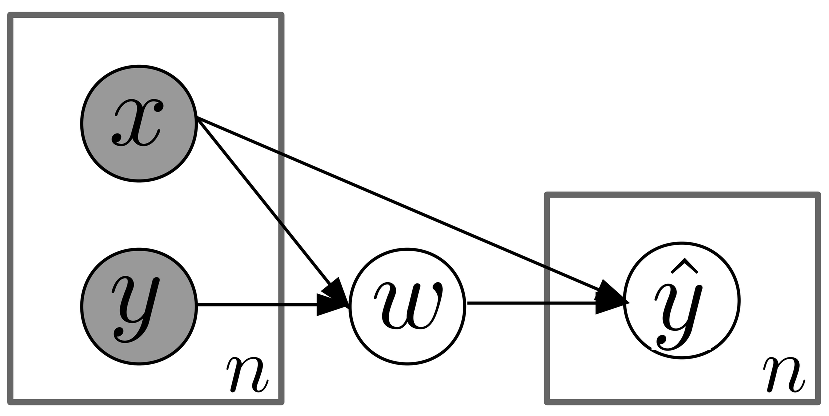
From a "Frequentist" viewpoint, we assume there exists a "true" *data distribution* $p^*(S)$, which is different from the *marginal likelihood* $p(S)$. Besides, we introduce an *encoder* $q(w|S)$, which is also different from the *posterior* $p(w|S)$. Then, the latent variable model of the data decomposes as

$$p(S, w) = p^*(S)q(w|S).$$

Rate-distortion theory for supervised learning

Taking $p(y|x, w)$ as the decoder, $q(w|S)$ as the encoder, we have a full view of supervised learning with model uncertainty:

$$\text{Predictive}: q(y \mid x, S) = \int p(y \mid x, w)q(w|S)\, \mathrm{d}w.$$



The weight w can be interpreted as the code of the autoencoder.
Inspired by rate-distortion, we have a compression-error tradeoff:

$$\min_{q(w|S)\in\Delta} \Big[I(w; S) \equiv \mathbb{E}_{p^*(S)}\mathbb{E}_{q(w|S)} \log \frac{q(w|S)}{\sum_S p^*(S)q(w|S)} \Big]$$

$$\text{s.t. } \mathbb{E}_{p^*(S)}\mathbb{E}_{q(w|S)}\Big[d(w, S) \equiv -\sum_{i=1}^n \log p(y_i|x_i, w) \Big] \leq D$$

Applying variational characterization [3], we obtain

$$I(w; S) \equiv \min_{m(w)\in\Delta} \mathbb{E}_{q(w|S)}\big[\log \frac{q(w|S)}{m(w)}\big].$$

The classical Blahut-Arimoto algorithm [4, 5] takes the following steps alternatively with β the Lagrangian multiplier:

$$q(w|S) = \frac{m(w)\exp(-\beta\, d(w, S))}{\int m(v)\exp(-\beta\, d(v, S))dv}$$

$$m(w) = \sum_S p^*(S)q(w|S)$$

Interpretation: $I(w; S)$ is a regularizer, which forces w to contain less information about a particular S; less memorization implies better generalization.

Training β-BNN: approximate Blahut-Arimoto

Since q and m are intractable, we use variational approximation:

q step: update the "posterior" by a parametric approximation

$$\begin{aligned}\theta(S) &= \arg\min_\theta \mathrm{KL}\big(q(w|\theta)\|q(w|S)\big)\\ &= \arg\min_\theta \mathrm{KL}\big(q(w|\theta)\|m(w)\big) + \beta\, \mathbb{E}_{q(w|\theta)}\big[d(w, S)\big]\end{aligned}$$

m step: update the "prior" by a Monte Carlo approximation

$m(w) \simeq \sum_S p^*(S)q(w|\theta(S)) \simeq \frac{1}{K}\sum_{k=1}^K q(w|\theta(B_k))$, where B_k is a bootstrap sample of size n_b drawn from the empirical distribution $p_S(x, y) = \frac{1}{n}\sum_{i=1}^n \delta(x_i = x)\delta(y_i = y)$.

Experiments on Colorful MNIST [6]

Algorithm	β^*	Accuracy
Vanilla BNN	$\frac{1}{n}$	90.05
Fixed-prior β-BNN	10^{-10}	95.86
β-BNN	10^{-5}	96.08
Online β-BNN	10^{-3}	97.12

Table 1. Classification results.

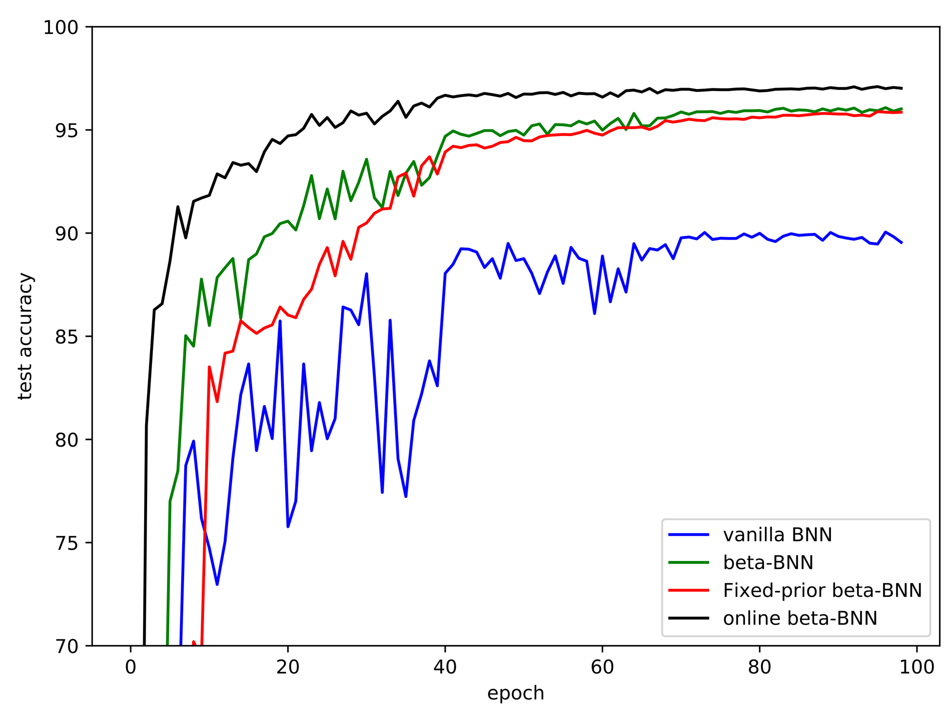


Figure 1. Test accuracy over epochs.

Experimental details:

- $q(w|\theta)$ is specified as a diagonal Gaussian.
- $p(y|x, w) = \mathrm{MLP}(x; w)$.
- q step is optimized by SGD with batch size 128, learning rate 10^{-3}.
- m step: $K = 5$, since the performance only increases marginally for $K \geq 5$.
- Bootstrap sample size $n_b = 10^4$. For *Online* β-*BNN*, $n_b = 128$.
- *Vanilla BNN* = *fixed-prior* β-*BNN* with $\beta = \frac{1}{n}$ and $K = 1$.

Bibliography

[1] Hinton, G. E. and Van Camp, D. (1993). Keeping the neural networks simple by minimizing the description length of the weights. In *COLT*, pages 5-13. ACM.

[2] Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight uncertainty in neural networks. In *ICML*.

[3] Cover, T. M. and Thomas, J. A. (2012). *Elements of information theory*.

[4] Blahut, R. (1972). Computation of channel capacity and rate-distortion functions. In *ISIT*, 18(4):460-473.

[5] Arimoto, S. (1972). An algorithm for computing the capacity of arbitrary discrete memoryless channels. In *ISIT*, 18(1):14-20.

[6] Bulten, W. (2017). https://bit.ly/2FYTnw3