Bayesian deep learning and uncertainty in deep learning

Seongok Ryu

ACE-Team, KAIST Chemistry

Statistical inference

Training a certain neural network model is equivalent to obtaining a posterior $p(\boldsymbol{\omega}|\mathbf{X},\mathbf{Y})$.

likelihood prior

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

evidence (or marginal likelihood)

In general, the model is obtained by solving optimization problem.

$$\mathcal{L}(\boldsymbol{\omega}) = \log p(\mathbf{Y}|\mathbf{X},\boldsymbol{\omega}) + \log p(\boldsymbol{\omega})$$

If we assume that the likelihood is Gaussian distribution $p(\mathbf{Y}|\mathbf{X},\boldsymbol{\omega}) \propto \exp\left(\frac{\left(\mathbf{Y}-\mathbf{f}^{\boldsymbol{\omega}}(\mathbf{X})\right)^2}{2\sigma^2}\right)$ and also the prior is Gaussian distribution $p(\boldsymbol{\omega}) \propto \boldsymbol{exp}\left(\frac{\|\boldsymbol{\omega}\|^2}{2l^2}\right)$, then the minimization objective is

$$\mathcal{L}(\boldsymbol{\omega}) = \left(\mathbf{Y}-\mathbf{f}^{\boldsymbol{\omega}}(\mathbf{X})\right)^2 + \frac{\|\boldsymbol{\omega}\|^2}{2l^2} + \text{const.}$$

L2-norm (MSE)

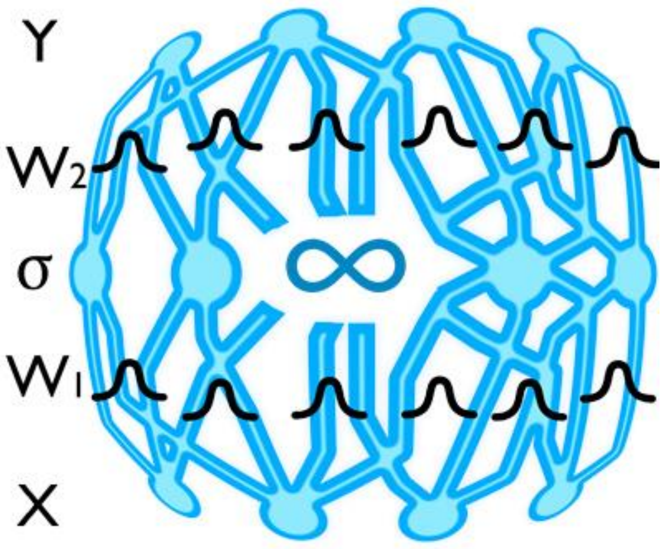
L2-regularization
l: prior length scale

Statistical inference

Bayesian's eye
: how to do inference about **hypotheses (uncertain quantities)** from **data (measured quantities)**.

$$p(\mathbf{hypothesis}|\mathbf{data}) = \frac{p(\mathbf{data}|\mathbf{hypothesis}) \times p(\mathbf{hypothesis})}{P(\mathbf{data})}$$

https://www.youtube.com/watch?v=FD8l2vPU5FY&t=2s



Bayesian let all quantities, except data, as uncertain quantities.

http://www.cs.ox.ac.uk/people/yarin.gal/website/blog_3d801aa532c1ce.html

Statistical inference

Frequentist inference: model parameters and predicted output is deterministic.

- Frequentist model estimation: $\hat{\boldsymbol{\omega}} = \underset{\boldsymbol{\omega}\in\Omega}{\operatorname{argmax}}\, \mathcal{L}(\boldsymbol{\omega})$
- Maximum-a-posteriori (MAP) estimation: $\hat{\boldsymbol{\omega}} = \underset{\boldsymbol{\omega}\in\Omega}{\operatorname{argmax}}\, p(\mathbf{Y}|\mathbf{X},\boldsymbol{\omega})\, p(\boldsymbol{\omega})$
- Maximum-likelihood estimation (MLE) : $\hat{\boldsymbol{\omega}} = \underset{\boldsymbol{\omega}\in\Omega}{\operatorname{argmax}}\, p(\mathbf{Y}|\mathbf{X},\boldsymbol{\omega})$

Assuming prior has a uniform distribution

- Frequentist inference: $\mathbf{y}^* = \mathrm{f}^{\hat{\boldsymbol{\omega}}}(\mathbf{x}^*)$

Bayesian inference: model parameters and predicted output are probabilistic (have distributions). In other words, it allows to model 'uncertainty' over parameters.

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

In this reason, we can estimate the uncertainty of our prediction by measuring the variance of predictive distribution $p(\mathbf{y}^*|\mathbf{x}^*,\mathbf{X},\mathbf{Y})$.

Statistical inference

Inference from maximum-a-posteriori (MAP) model estimation

Choosing the posterior
which can explain the given data distribution the best.

$$p(\mathrm{y}^*|\mathrm{x}^*) = \mathrm{f}^{\hat{\mathrm{w}}}(\mathrm{x}^*) \longleftarrow \hat{\mathrm{w}} = \underset{\mathrm{w}\in\mathcal{W}}{\operatorname{argmax}}\, p(\mathrm{w}|\mathrm{X},\mathrm{Y})$$

Then, our inference will be a single deterministic value.

Inference from Bayesian model estimation

Summation over all possible model posteriors

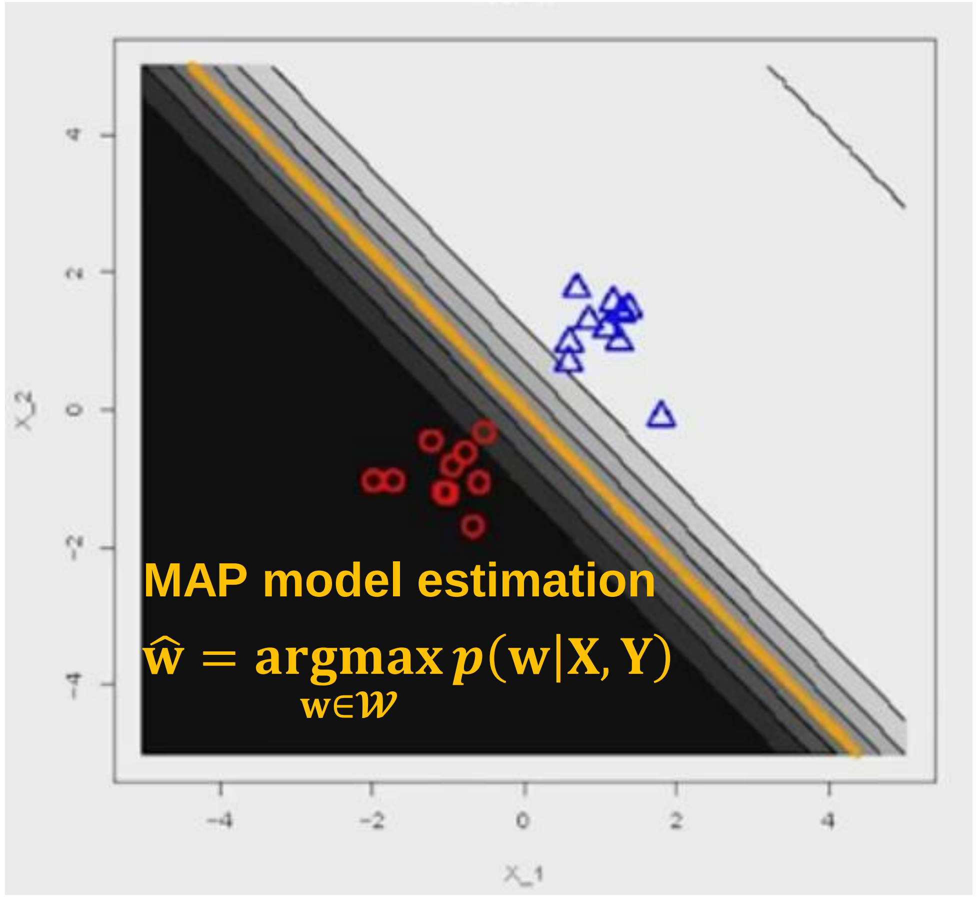
$$p(\mathrm{y}^*|\mathrm{x}^*,\mathrm{X},\mathrm{Y}) = \int_{w\in\mathcal{W}} p(\mathrm{y}^*|\mathrm{x}^*,\mathrm{w})p(\mathrm{w}|\mathrm{X},\mathrm{Y})d\mathrm{w}$$

Then, our inference will have a distribution instead of a single deterministic value.

→ **The uncertainty on model parameter give the uncertainty on our inference**

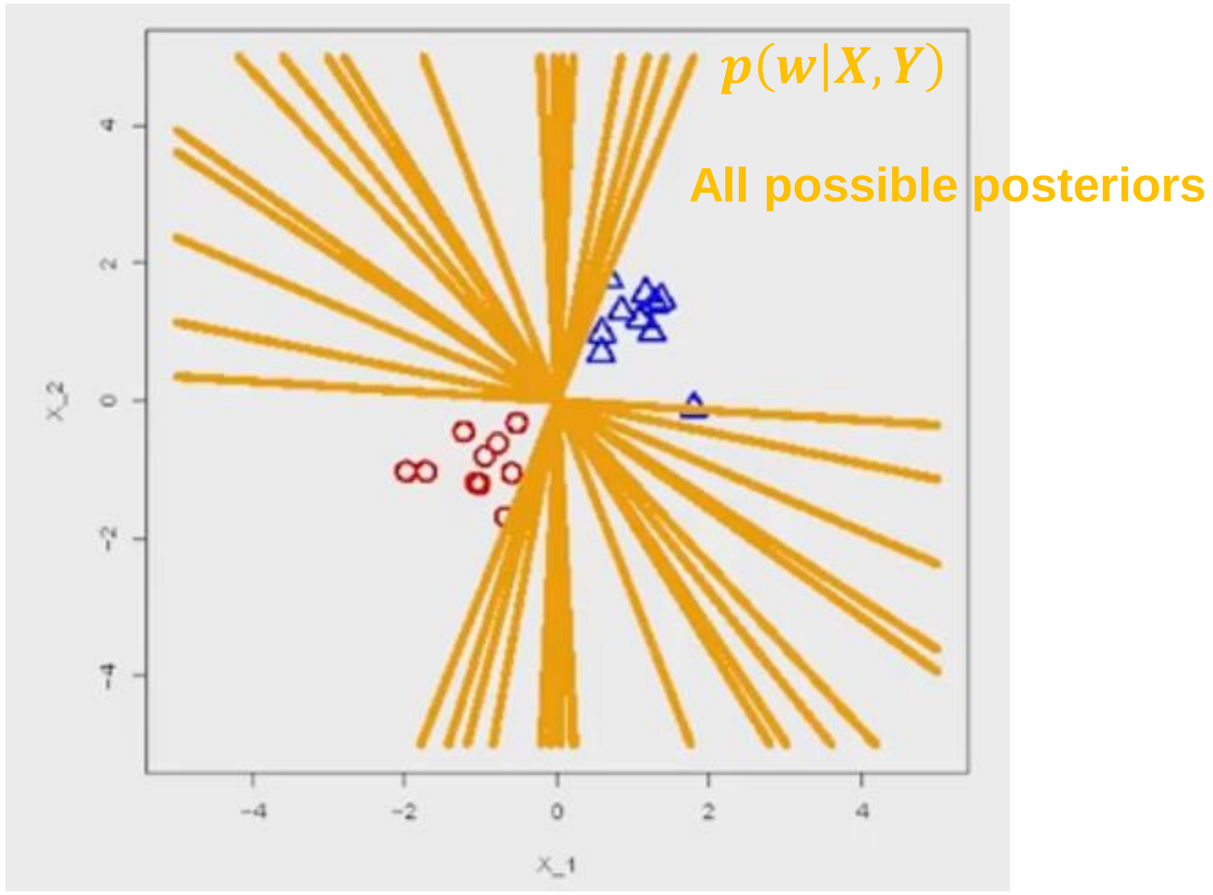
Statistical inference

MAP predictive distribution



$$p(y^*|x^*) = f^{\hat{w}}(x^*)$$

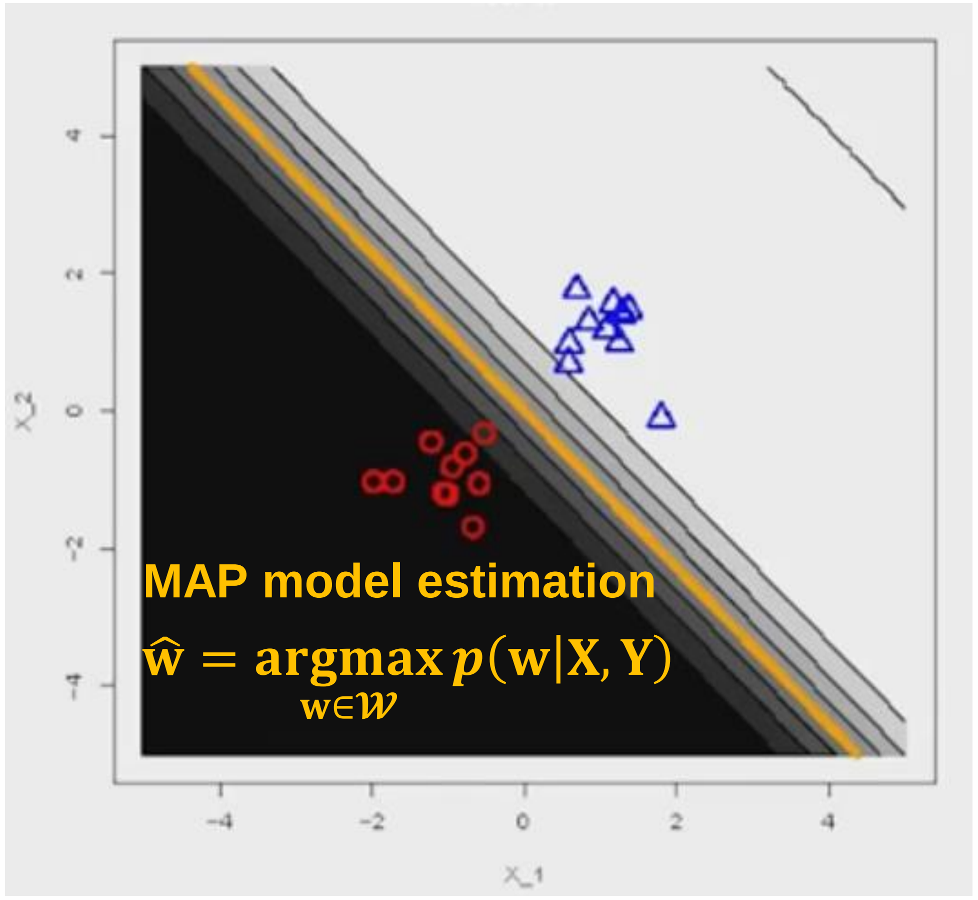
Bayesian predictive distribution



https://www.youtube.com/watch?v=1ClZhMSHeBA

Statistical inference

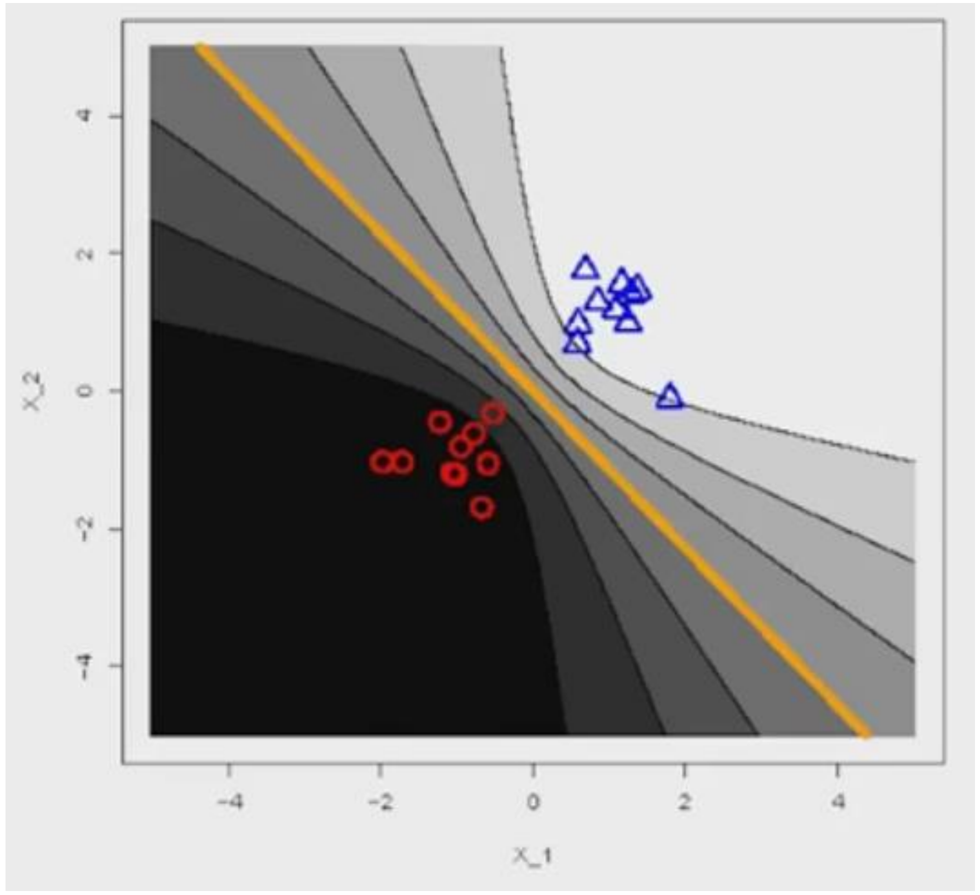
MAP predictive distribution



MAP model estimation

$$\hat{\mathrm{w}} = \operatorname*{argmax}_{\mathrm{w} \in \mathcal{W}} p(\mathrm{w}|\mathrm{X}, \mathrm{Y})$$

Bayesian predictive distribution



$$p(\mathrm{y}^*|\mathrm{x}^*) = \mathrm{f}^{\hat{\mathrm{w}}}(\mathrm{x}^*)$$

$$p(\mathrm{y}^*|\mathrm{x}^*, \mathrm{X}, \mathrm{Y}) = \int_{w \in \mathcal{W}} p(\mathrm{y}^*|\mathrm{x}^*, \mathrm{w}) p(\mathrm{w}|\mathrm{X}, \mathrm{Y}) dw$$

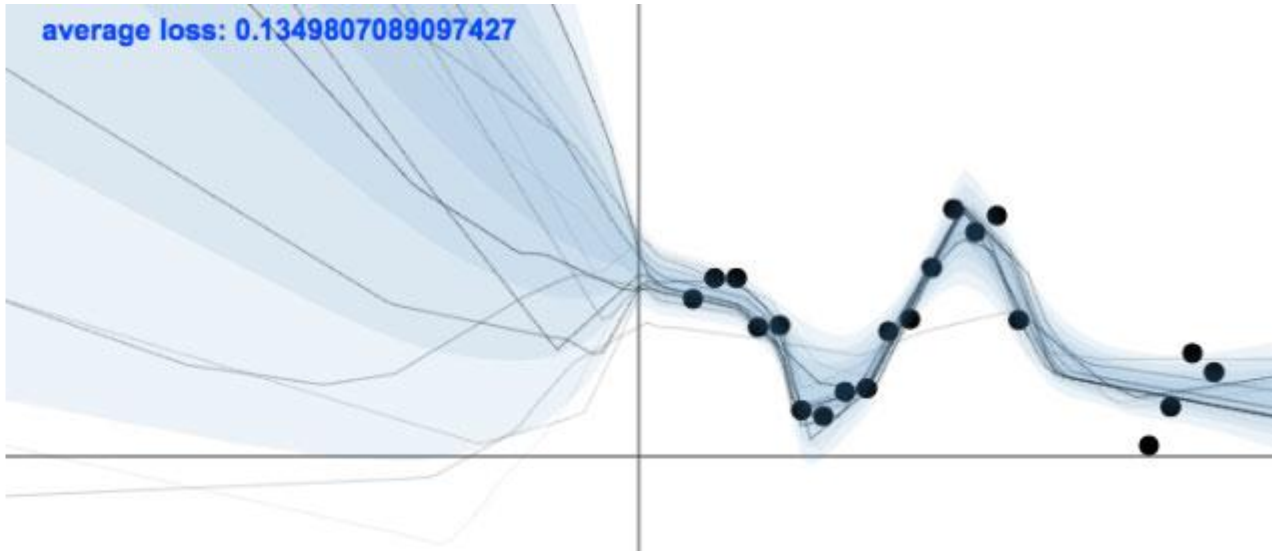
https://www.youtube.com/watch?v=1ClZhMSHeBA

Statistical inference

Bayesian inference and uncertainty

We can estimate **the uncertainty** on **our inference** as variance of predictive distribution

$$Var[p(\mathbf{y}^*|\mathbf{x}^*,\mathbf{X},\mathbf{Y})] \longleftarrow p(\mathbf{y}^*|\mathbf{x}^*,\mathbf{X},\mathbf{Y}) = \int_{w\in\mathcal{W}} p(\mathbf{y}^*|\mathbf{x}^*,\mathbf{w})p(\mathbf{w}|\mathbf{X},\mathbf{Y})d\mathbf{w}$$

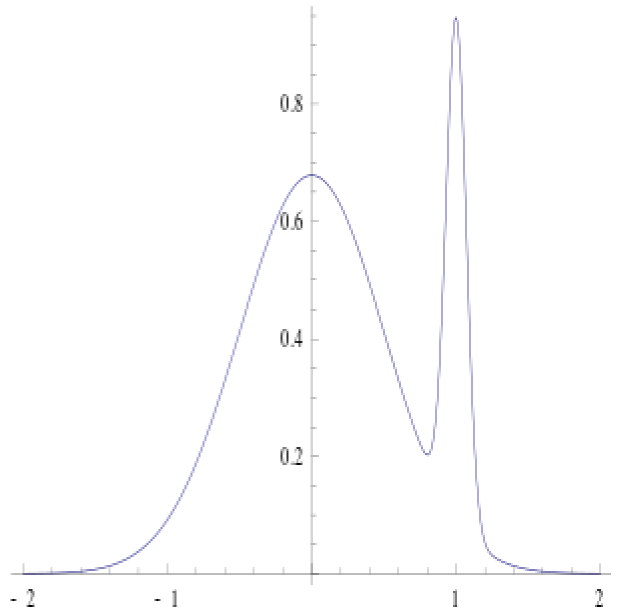


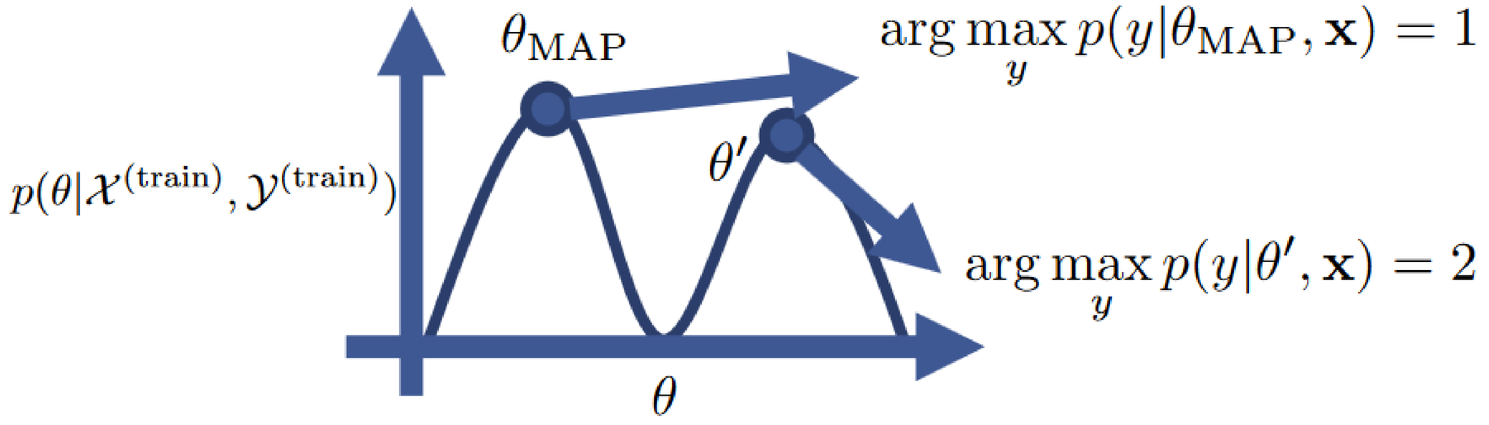
"Knowing what we don't know" **is as important as** ***"knowing itself".***

http://www.cs.ox.ac.uk/people/yarin.gal/website/blog_2248.html

Statistical inference

Limitations of Frequentist’s model estimation:





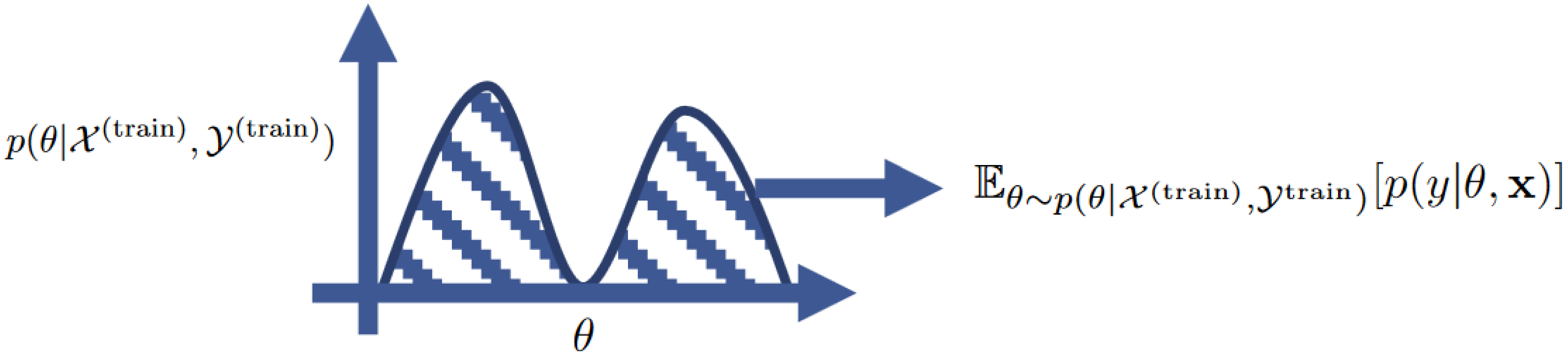
θ_{MAP}
$\arg\max_y p(y|\theta_{\text{MAP}}, \mathbf{x}) = 1$
θ'
$p(\theta|\mathcal{X}^{(\text{train})}, \mathcal{Y}^{(\text{train})})$
$\arg\max_y p(y|\theta', \mathbf{x}) = 2$
θ

✓ MAP estimation only considers a single point estimate:
It may fails when the posterior have a multi-modal distribution in which the highest mode is uncharacteristic of the majority of the distribution.

Statistical inference

Limitations of Bayesian's model estimation:

Bayesian inference: model parameters and predicted output are probabilistic (have distributions).


$p(\theta|\mathcal{X}^{(\text{train})}, \mathcal{Y}^{(\text{train})})$
θ
$\mathbb{E}_{\theta \sim p(\theta|\mathcal{X}^{(\text{train})}, \mathcal{Y}^{\text{train}})}[p(y|\theta, \mathbf{x})]$

http://alinlab.kaist.ac.kr/resource/Lec8_Bayesian_DL.pdf

Statistical inference

Limitations of Bayesian's model estimation:

Bayesian inference: model parameters and predicted output are probabilistic (have distributions).

$$p(\boldsymbol{\omega}|\mathbf{X},\mathbf{Y}) = \frac{\overbrace{p(\mathbf{Y}|\mathbf{X},\boldsymbol{\omega})}^{\text{likelihood}}\overbrace{p(\boldsymbol{\omega})}^{\text{prior}}}{\underbrace{p(\mathbf{Y}|\mathbf{X})}_{\text{evidence (or marginal likelihood)}}}$$

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

✓ Choice of prior

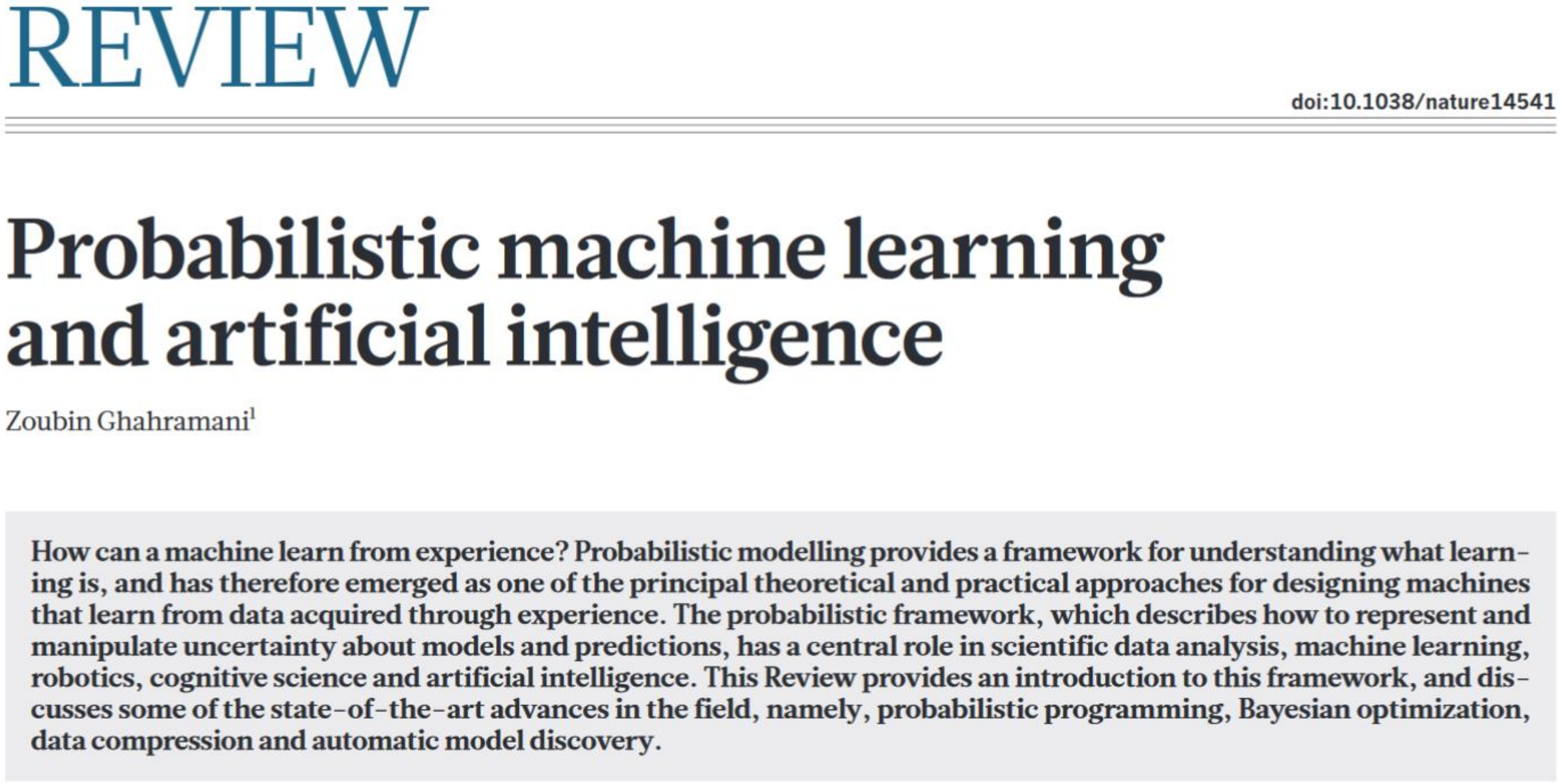
: Is assuming prior as Gaussian distribution correct? How can we choose a good prior?

✓ Posterior is usually intractable

: good approximation is needed, but it often hurts the quality of approximated posterior.

Statistical inference

Recommend to read the review article written by Zoubin Ghahramani.
I think that this article is good for beginners.



REVIEW

doi:10.1038/nature14541

Probabilistic machine learning and artificial intelligence

Zoubin Ghahramani[1]

How can a machine learn from experience? Probabilistic modelling provides a framework for understanding what learning is, and has therefore emerged as one of the principal theoretical and practical approaches for designing machines that learn from data acquired through experience. The probabilistic framework, which describes how to represent and manipulate uncertainty about models and predictions, has a central role in scientific data analysis, machine learning, robotics, cognitive science and artificial intelligence. This Review provides an introduction to this framework, and discusses some of the state-of-the-art advances in the field, namely, probabilistic programming, Bayesian optimization, data compression and automatic model discovery.

Bayesian inference

A neural network model, which is consisted of trainable parameters ω
and trained with training data set $D = \{X, Y\}$

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

prior

posterior

Bayesian model

1. Model parameters are not deterministic but probabilistic.
2. Model outputs are not deterministic but have distributions.

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) = \int_{\Omega} p(\mathbf{y}^*|\mathbf{x}^*, \boldsymbol{\omega})p(\boldsymbol{\omega}|\mathbf{X}, \mathbf{Y})d\boldsymbol{\omega}$$

Bayesian inference

However, using above formulation directly isn't practical, because

- the posterior is intractable.
- the integration of parameters over whole parameter space Ω is also impossible.

Approximations in Bayesian modeling

Today's topic

1. **Variational inference: $p(\boldsymbol{\omega}|\mathbf{X},\mathbf{Y}) \approx q_{\theta}(\boldsymbol{\omega})$**

   - Fully Factorized Gaussian (FFG), also referred as mean-field approximation
   Blundell, Charles, et al. "Weight uncertainty in neural networks." *arXiv preprint arXiv:1505.05424* (2015).
   - Multiplicative Normalizing Flow (MNF)
   Louizos, Christos, and Max Welling. "Multiplicative normalizing flows for variational bayesian neural networks." *arXiv preprint arXiv:1703.01961* (2017).
   - Dropout network
   Gal, Yarin, and Zoubin Ghahramani. "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning." *international conference on machine learning*. 2016.
   - …

2. **Laplace approximation: pointwise estimation assisted with posterior curvature.**

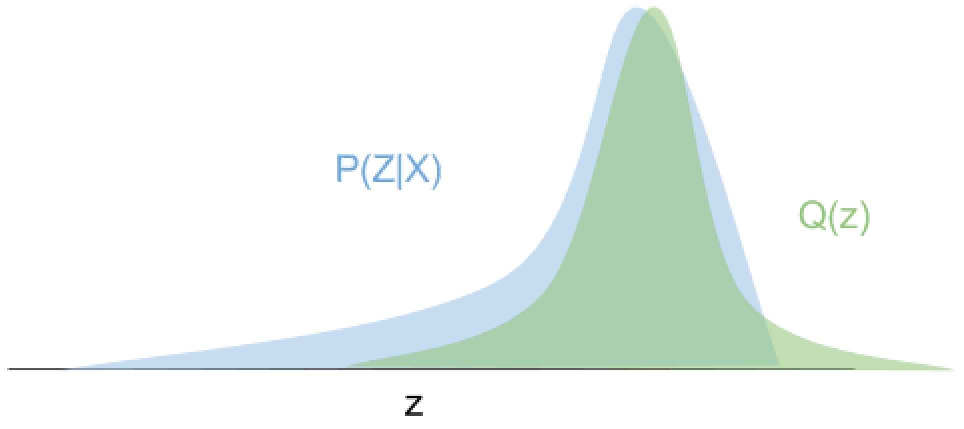
3. **Markov chain Monte Carlo (MCMC): running Markov chains for Monte Carlo estimate of the posterior.**

Variational inference

Reparameterization trick using variational parameter θ

$$p(\boldsymbol{\omega}|\mathbf{X},\mathbf{Y}) \approx q_\theta\,(\boldsymbol{\omega})$$

Kullback-Leibeler divergence
: metric to make two distributions similar



$$\mathrm{KL}\big(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega}|\mathbf{X},\mathbf{Y})\big) = \int_\Omega q_\theta(\boldsymbol{\omega}) \log \frac{q_\theta(\boldsymbol{\omega})}{p(\boldsymbol{\omega}|\mathbf{X},\mathbf{Y})} d\boldsymbol{\omega}$$

Still intractable because the posterior exists in the KL term.

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

$$\mathcal{L}_{VI}(\omega) = -\int_\Omega q_\theta(\boldsymbol{\omega}) \log p(\mathbf{Y}|\mathbf{X},\boldsymbol{\omega}) d\boldsymbol{\omega} + \mathrm{KL}\big(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega})\big)$$

Minimization objective is equivalent to minimizing the evidence lower-bound (ELBO).

Variational inference

Therefore, our minimization objective is

$$\begin{aligned}\mathrm{KL}\big(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega}|\mathbf{X},\mathbf{Y})\big) &= -\int_\Omega q_\theta(\boldsymbol{\omega})\log p(\mathbf{y}_i|\mathbf{f}^{\boldsymbol{\omega}}(\mathbf{x}_i))d\boldsymbol{\omega} + \mathrm{KL}\big(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega})\big)\\ &= -\sum_{i=1}^{N}\int_\Omega q_\theta(\boldsymbol{\omega})\log p(\mathbf{y}_i|\mathbf{f}^{\boldsymbol{\omega}}(\mathbf{x}_i))d\boldsymbol{\omega} + \mathrm{KL}\big(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega})\big)\end{aligned}$$

Instead of performing computations over the entire dataset, we may use data sub-sampling, also referred to as mini-batch optimization.

$$\mathrm{KL}\big(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega}|\mathbf{X},\mathbf{Y})\big) = -\frac{N}{M}\sum_{i\in S}^{M}\int_\Omega q_\theta(\boldsymbol{\omega})\log p(\mathbf{y}_i|\mathbf{f}^{\boldsymbol{\omega}}(\mathbf{x}_i))d\boldsymbol{\omega} + \mathrm{KL}\big(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega})\big)$$

$$\to \frac{1}{M}\sum_{i\in S}^{M}\int_\Omega q_\theta(\boldsymbol{\omega})\log p(\mathbf{y}_i|\mathbf{f}^{\boldsymbol{\omega}}(\mathbf{x}_i))d\boldsymbol{\omega} + \frac{1}{N}\mathrm{KL}\big(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega})\big)$$

Gal, Yarin. "Uncertainty in deep learning." *University of Cambridge* (2016).

Reparameterization trick

For practical applications, the integration over whole parameter space can be replaced to summation of subsampled parameters with a Monte Carlo (MC) estimator.

$$\mathcal{L}_{VI}(\theta) = -\frac{1}{M}\sum_{i\in S}\int_{\Omega} \log p(\mathbf{y}_i|\mathbf{f}^{\boldsymbol{\omega}}(\mathbf{x}_i))\, q_\theta(\boldsymbol{\omega})d\boldsymbol{\omega} + \frac{1}{N}\mathrm{KL}(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega}))$$

$$= -\frac{1}{M}\sum_{i\in S}\int_{\Omega} \log p\left(\mathbf{y}_i\middle|\mathbf{f}^{g(\theta,\epsilon)}(\mathbf{x}_\mathrm{i})\right) p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon} + \frac{1}{N}KL(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega}))$$

$$\hat{\mathcal{L}}_{MC}(\theta) = -\frac{1}{M}\sum_{i\in S}\log p\left(\mathbf{y}_i\middle|\mathbf{f}^{g(\theta,\epsilon)}(\mathbf{x}_i)\right) + \frac{1}{N}\mathrm{KL}(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega}))$$

We can then estimate the predictive distribution with MC integration as well.

$$\tilde{q}_\theta(\mathbf{y}^*|\mathbf{x}^*) := \frac{1}{T}\sum_{t=1}^{T} p(\mathbf{y}^*|\mathbf{x}^*, \widehat{\boldsymbol{\omega}}_t) \xrightarrow[T\to\infty]{} \int p(\hat{\mathbf{y}}^*|\hat{\mathbf{x}}^*, \boldsymbol{\omega})q_\theta(\boldsymbol{\omega})d\boldsymbol{\omega}$$

$$\approx \int p(\hat{\mathbf{y}}^*|\hat{\mathbf{x}}^*, \boldsymbol{\omega})p(\boldsymbol{\omega}|\mathbf{X},\mathbf{Y})d\boldsymbol{\omega}$$

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

Gal, Yarin. "Uncertainty in deep learning." *University of Cambridge* (2016).

Dropout network

- **Dropout as one of the stochastic regularization techniques**

In Bayesian neural networks, the stochasticity comes from **our uncertainty over the model parameters**.

We can transform dropout's noise from the feature space to the parameter space as follows.

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

writing $\widehat{\mathbf{W}}_1 \coloneqq \mathrm{diag}(\hat{\boldsymbol{\epsilon}}_1)\mathbf{M}_1$ and $\widehat{\mathbf{W}}_2 \coloneqq \mathrm{diag}(\hat{\boldsymbol{\epsilon}}_2)\mathbf{M}_2$, we end up with

$$\mathbf{y} = \sigma\big(\mathbf{x}\widehat{\mathbf{W}}_1 + \mathbf{b}\big)\widehat{\mathbf{W}}_2 \eqqcolon \mathbf{f}^{\widehat{\mathbf{W}}_1, \widehat{\mathbf{W}}_2, \mathbf{b}}(\mathbf{x})$$

with random variable realizations as weights, and write $\widehat{\boldsymbol{\omega}} = \{\widehat{\mathbf{W}}_1, \widehat{\mathbf{W}}_2, \mathbf{b}\}$

Gal, Yarin. "Uncertainty in deep learning." *University of Cambridge* (2016).

Dropout network

$$\mathbf{y} = \sigma\left(\mathbf{x}\widehat{\mathbf{W}}_1 + \mathbf{b}\right)\widehat{\mathbf{W}}_2 + \mathbf{b}_2 =: \mathbf{f}^{\widehat{\mathbf{W}}_1, \widehat{\mathbf{W}}_2, \mathbf{b}}(\mathbf{x})$$

This allows us to write dropout's objective in a more convenient form.

$$\hat{\mathcal{L}}_{dropout}(\mathbf{M}_1, \mathbf{M}_2, \mathrm{b}) \coloneqq \frac{1}{M}\sum_{i \in S} E^{\widehat{\mathbf{W}}_1, \widehat{\mathbf{W}}_2, \mathbf{b}}(\mathbf{x}_i, \mathbf{y}_i) + \lambda_1 \|\mathbf{M}_1\|^2 + \lambda_2 \|\mathbf{M}_2\|^2 + \lambda_3 \|\mathbf{b}\|^2$$

$$E^{\widehat{\mathbf{W}}_1, \widehat{\mathbf{W}}_2, \mathbf{b}}(\mathbf{x}, \mathbf{y}\,) = \frac{1}{2}\left\|\mathrm{y} - \mathbf{f}^{\widehat{\mathbf{W}}_1, \widehat{\mathbf{W}}_2, \mathbf{b}}(\mathbf{x})\right\|^2 = -\frac{1}{\tau}\log p\left(\mathbf{y}|\mathbf{f}^{\widehat{\mathbf{W}}_1, \widehat{\mathbf{W}}_2, \mathbf{b}}(\mathbf{x})\right) + \mathrm{const}$$

where $p\left(\mathbf{y}|\mathbf{f}^{\widehat{\mathbf{W}}_1, \widehat{\mathbf{W}}_2, \mathbf{b}}(\mathbf{x})\right) = \mathcal{N}\left(\mathbf{y}; \mathbf{f}^{\widehat{\mathbf{W}}_1, \widehat{\mathbf{W}}_2, \mathbf{b}}(\mathbf{x}), \tau^{-1} I\right)$ with τ^{-1} observation noise.

Gal, Yarin. "Uncertainty in deep learning." *University of Cambridge* (2016).

Dropout network

Recall that $\widehat{\boldsymbol{\omega}} = \{\widehat{\mathbf{W}}_1, \widehat{\mathbf{W}}_2, \mathbf{b}\}$ and write

$$\widehat{\boldsymbol{\omega}}_{\boldsymbol{i}} = \{\widehat{\mathbf{W}}_1^i, \widehat{\mathbf{W}}_2^i, \mathbf{b}^{\boldsymbol{i}}\} = \{\mathrm{diag}(\hat{\boldsymbol{\epsilon}}_1^i)\mathbf{M}_1, \mathrm{diag}(\hat{\boldsymbol{\epsilon}}_2^i)\mathbf{M}_2, \mathbf{b}\} =: g(\theta, \hat{\boldsymbol{\epsilon}}_i)$$

with $\theta = \{\mathbf{M}_1, \mathbf{M}_2, \mathbf{b}\}, \hat{\boldsymbol{\epsilon}}_1^i \sim p(\boldsymbol{\epsilon}_1)$, and $\hat{\boldsymbol{\epsilon}}_2^i \sim p(\boldsymbol{\epsilon}_2)$ for $1 \le i \le N$. Here $p(\boldsymbol{\epsilon}_l)$ $(l = 1,2)$ is a product of Bernoulli distributions with probabilities $1 - p_l$, from which a realization would be a vector of zeros and ones.

We can get objective

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

with $\hat{\boldsymbol{\epsilon}}_i$ realizations of the random variable $\boldsymbol{\epsilon}$.

Gal, Yarin. "Uncertainty in deep learning." *University of Cambridge* (2016).

Dropout network

Compare two objectives,

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

and

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

If we define the prior $p(\boldsymbol{\omega})$ s.t. the following holds:

$$\frac{\partial}{\partial\theta}\mathrm{KL}\left(q_\theta(\boldsymbol{\omega})\middle\|p(\boldsymbol{\omega})\right) = \frac{\partial}{\partial\theta}N\tau(\lambda_1\|\mathbf{M}_1\|^2 + \lambda_2\|\mathbf{M}_2\|^2 + \lambda_3\|\mathbf{b}\|^2)$$

(**referred as the KL condition**), we would have the following relation between the derivatives of two objectives

$$\frac{\partial}{\partial\theta}\hat{\mathcal{L}}_{dropout}(\theta) = \frac{1}{N\tau}\frac{\partial}{\partial\theta}\hat{\mathcal{L}}_{MC}(\theta)$$

with identical optimization procedure.

Gal, Yarin. "Uncertainty in deep learning." *University of Cambridge* (2016).

Uncertainties

Homoskedastic regression assumes constant noise σ for every input point $\mathbf{x}$.

On the other hand, we draw model weights from the approximate posterior $\hat{\boldsymbol{\omega}} \sim q(\boldsymbol{\omega})$ to obtain a model output, this time composed of both predictive mean as well as predictive variance:

$$[\hat{\mathbf{y}}_i, \hat{\sigma}_i{}^2] = \mathbf{f}^{\hat{\boldsymbol{\omega}}}(\mathbf{x_i})$$

Heteroskedastic regression, on the other hand, assumes that observation noise can vary with input $\mathbf{x}$.

We fix a Gaussian likelihood to model our aleatoric uncertainty. This induces a minimization objective given labeled output points $\mathbf{x}$:

$$\mathcal{L}_{NN}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{2\sigma_i^2} \|\mathbf{y}_i - \hat{\mathbf{y}}_i\|^2 + \frac{1}{2}\log \sigma_i^2$$

To summarize, the predictive uncertainty can be approximated using:

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

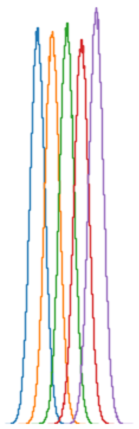
Epistemic uncertainty **Aleatoric uncertainty**

Kendall, Alex, and Yarin Gal.
"What uncertainties do we need in bayesian deep learning for computer vision?."
Advances in neural information processing systems. 2017.

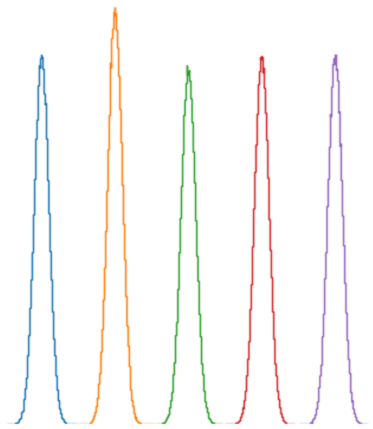
$$Var(y) \approx \frac{1}{T}\sum_{t=1}^{T}\hat{y}_t^2 - \left(\frac{1}{T}\sum_{t=1}^{T}\hat{y}_t\right)^2 + \frac{1}{T}\sum_{t=1}^{T}\hat{\sigma}_t^2$$

Uncertainties

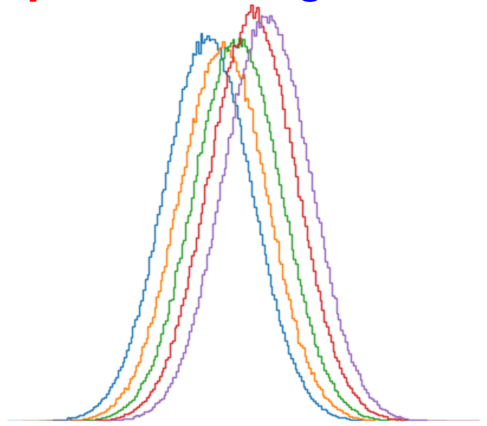
Low epistemic, Low aleatoric



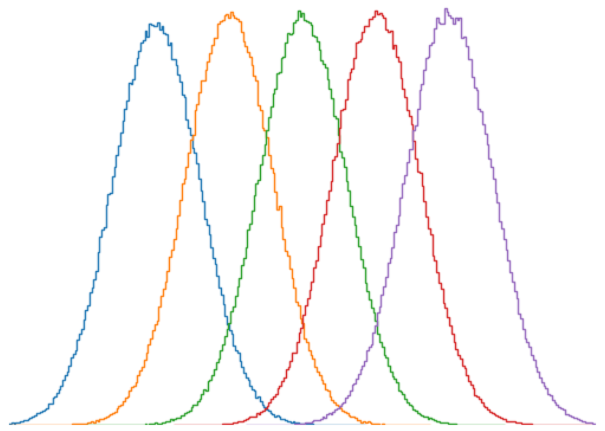
High epistemic, Low aleatoric



Low epistemic, High aleatoric



High epistemic, High aleatoric



Kendall, Alex, and Yarin Gal. "What uncertainties do we need in bayesian deep learning for computer vision?." *Advances in neural information processing systems*. 2017.

Summary and Questions

- What enables the Bayesian neural network to turn out predictive distributions?
  → Stochasticity of model parameters (uncertainty over the model parameters)

- How do we approximate the posterior, which is intractable?
  → Variational inference, Reparameterization with i) Dropout as Bayesian approximation, ii) Monte-Carlo estimator

- How do we measure the uncertainties of outcomes?
  → Variance of MC-sampled predictive mean (epistemic uncertainty) and mean of variance of MC-sampled predictive distributions

- Is the aleatoric uncertainty reducible as the number of training samples increases?
  → In principle, no.

- Is the epistemic uncertainty reducible as the number of training samples increases?
  → In principle, yes. The epistemic uncertainty is often referred as ‘reducible uncertainty’.

Summary and Questions

- How does dropout probability change with increasing the amount of training data?
  → The dropout probability will be reduced.
  When we train the model with few training data, the model regularization term ought to be large (ought to have small model capacity). In other words, the dropout probability ought to be large.
  → On the other hand, when we increase the amount of data, the model becomes able to have large model capacity and the dropout probability will be reduced.

- How can it be possible?
  $\rightarrow \frac{1}{M}\sum_{i\in S}^{M}\int_{\Omega} q_\theta(\boldsymbol{\omega})\log p(\mathbf{y}_i|\mathbf{f}^{\boldsymbol{\omega}}(\mathbf{x}_i))d\boldsymbol{\omega} + \frac{\mathbf{1}}{\boldsymbol{N}}\mathrm{KL}\big(q_\theta(\boldsymbol{\omega})||p(\boldsymbol{\omega})\big)$

- In other words, we have to find the individual optimal dropout probability of models trained with different amount of data.
  → Also, the dropout probability of each layer may be different.
  → Finding them manual grid-searching is impossible.
  → The solution what I have used is **"Concrete dropout".**

Concrete Dropout

- When using the dropout neural networks (or any other stochastic regularization technique), a randomly drawn masked weight matrix corresponds to a function draw.
- Therefore, the dropout probability, together with the weight configuration of the network, determine the magnitude of the epistemic uncertainty.
- **For a fixed dropout probability p,** high magnitude weights will result in higher output variance, i.e., higher epistemic uncertainty.
- With a fixed p, a model wanting to decrease its epistemic uncertainty will have to reduce its weight magnitude (and set the weights to be exactly zero to have zero epistemic uncertainty). Of course, this is impossible, as the model will not be able to explain the data well with zero weight matrices, therefore some balance between desired output variance and weight magnitude is achieved.
- **Allowing the probability to change** will let the model decrease its epistemic uncertainty by choosing smaller dropout probabilities.
- But if we wish to replace the grid-search with a gradient method, we need to define an optimization objective to optimize p with respect to.
- This is not trivial thing, as our aim is not to maximize model performance, but rather to obtain *good epistemic uncertainty*. What is a suitable objective for this?

Concrete Dropout

Recall that our minimization objective for the Dropout-based Bayesian neural network is

$$\hat{\mathcal{L}}_{MC}(\theta) = \frac{1}{M}\sum_{i\in S}^{M}\int_{\Omega} q_\theta(\boldsymbol{\omega}) \log p(\mathbf{y}_i|\mathbf{f}^{\boldsymbol{\omega}}(\mathbf{x}_i))d\boldsymbol{\omega} + \frac{1}{N}\mathrm{KL}\big(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega})\big),$$

where $\theta = \{\mathbf{M}_l, p_l\}_{l=1}^{L}$, $q_\theta(\boldsymbol{\omega}) = \Pi_l q_{\mathbf{M}_l}(\mathbf{W}_l)$ and $q_{\mathbf{M}_l}(\mathbf{W}_l) = \mathbf{M}_l \cdot \mathrm{diag}[\mathrm{Bernoulli}(1-p_l)^{K_l}]$.

If the dropout probabilities $\{p_l\}_{l=1}^{L}$ also become learnable parameters, the regularization term which is optimized to satisfy the KL-condition is given by

$$\mathrm{KL}\big(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega})\big) = \sum_{l=1}^{L}\mathrm{KL}\left(q_{\mathbf{M}_l}(\mathbf{W}_l)\|p(\mathbf{W}_l)\right)$$

$$\mathrm{KL}\left(q_{\mathbf{M}_l}(\mathbf{W}_l)\|p(\mathbf{W}_l)\right) \propto \frac{l^2(1-p_l)}{2}\,\|\mathbf{M}\|^2 - K\mathcal{H}(p_l)$$

with

$$\mathcal{H}(p_l) = -p_l \log p_l - (1-p_l)\log(1-p_l)$$

the *entropy* of a Bernoulli random variable with probability p_l.

Gal, Yarin, Jiri Hron, and Alex Kendall. "Concrete dropout." *Advances in Neural Information Processing Systems*. 2017.

Concrete Dropout

$$\hat{\mathcal{L}}_{MC}(\theta) = \frac{1}{M}\sum_{i\in S}^{M}\int_{\Omega} q_\theta(\boldsymbol{\omega})\log p(\mathbf{y}_i|\mathbf{f}^{\boldsymbol{\omega}}(\mathbf{x}_i))d\boldsymbol{\omega} + \frac{1}{N}\mathrm{KL}\big(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega})\big),$$

$$\mathrm{KL}\big(q_\theta(\boldsymbol{\omega})\|p(\boldsymbol{\omega})\big) = \sum_{l=1}^{L}\mathrm{KL}\left(q_{\mathbf{M}_l}(\mathbf{W}_l)\|p(\mathbf{W}_l)\right)$$

$$\mathrm{KL}\left(q_{\mathbf{M}_l}(\mathbf{W}_l)\|p(\mathbf{W}_l)\right) \propto \frac{l^2(1-p_l)}{2}\|\mathbf{M}\|^2 - K\mathcal{H}(p_l)$$

with

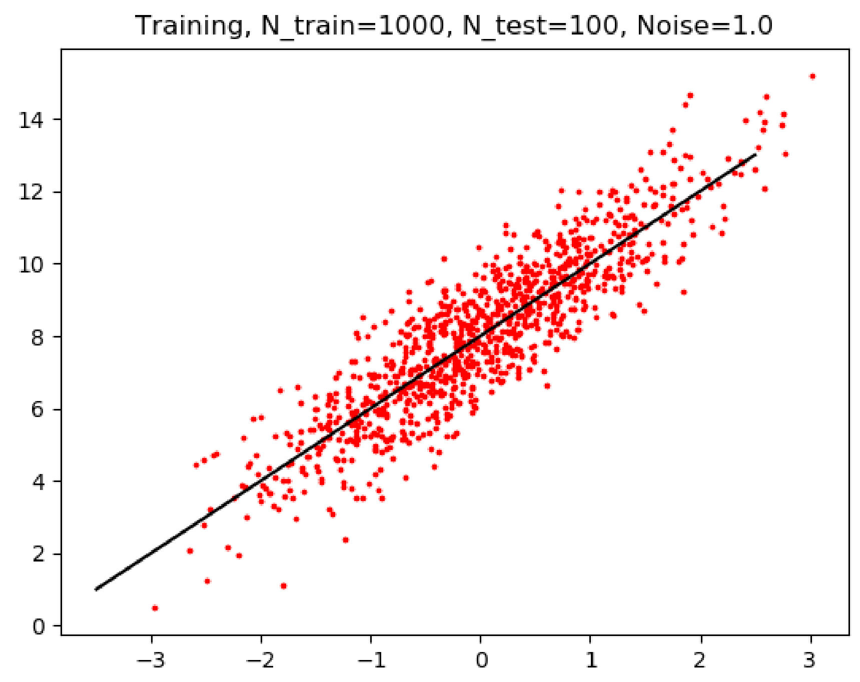
$$\mathcal{H}(p_l) = -p_l\log p_l - (1-p_l)\log(1-p_l)$$

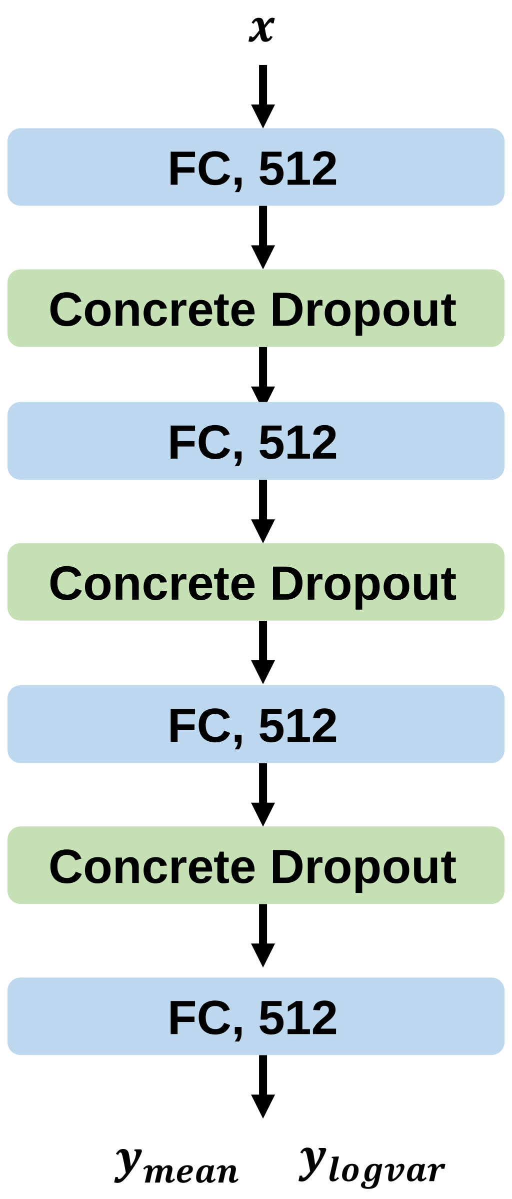
“The entropy term can be seen as a ***dropout regularization*** term. Minimizing the KL-divergence is equivalent to maximizing the entropy. This pushes the dropout probability towards 0.5 – the highest it can attain. The scailing of the regularization term means that large models will push the dropout probability towards 0.5 much more than smaller models, but as the amount of data N increases the dropout probability will be pushed towards 0.”

Gal, Yarin, Jiri Hron, and Alex Kendall. "Concrete dropout." *Advances in Neural Information Processing Systems*. 2017.

Toy model

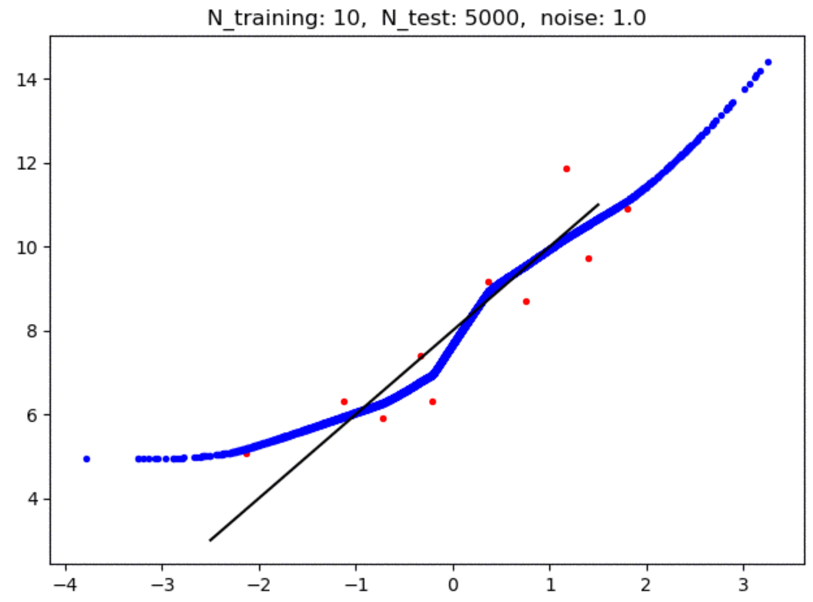
- Toy model) $y = 2x + 8 + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2)$
- Varying
  1) Number of training samples
  2) Amount of random Gaussian noise σ^2



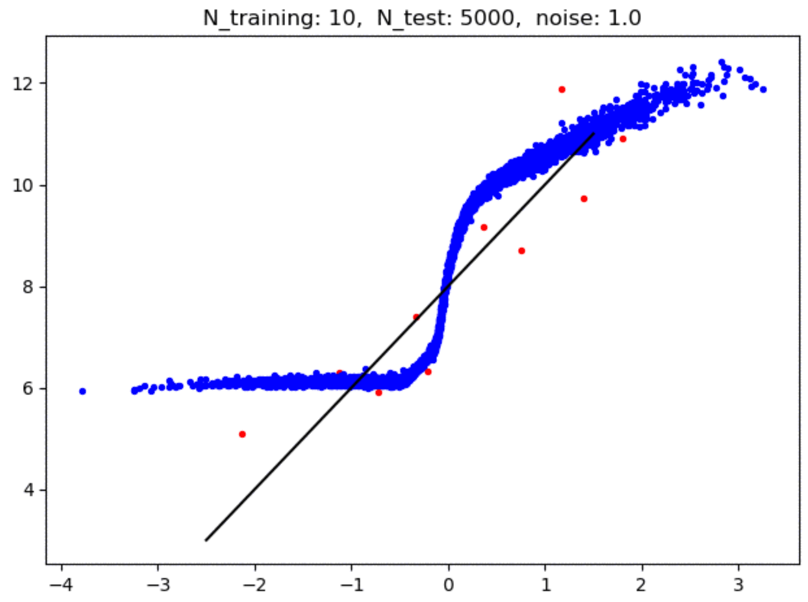


Toy model

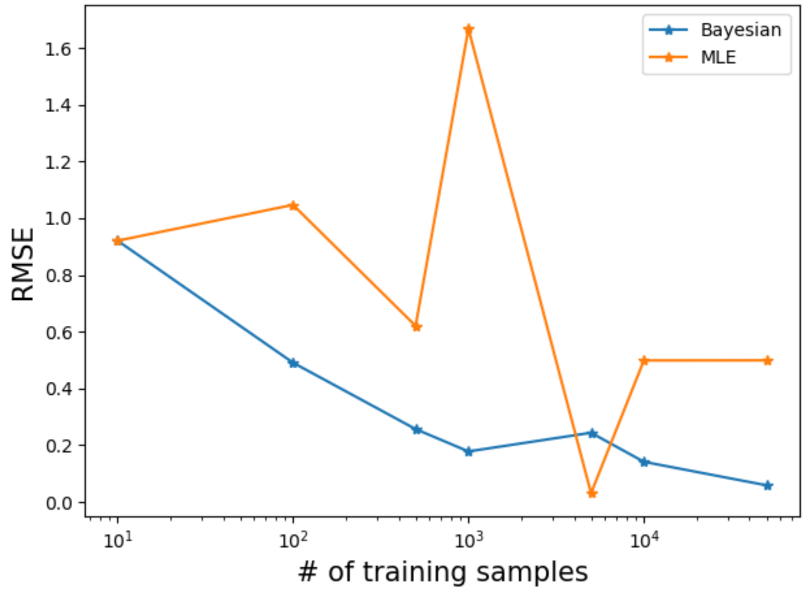
1. The amount of training data increases



Maximum Likelihood Estimation (MLE)



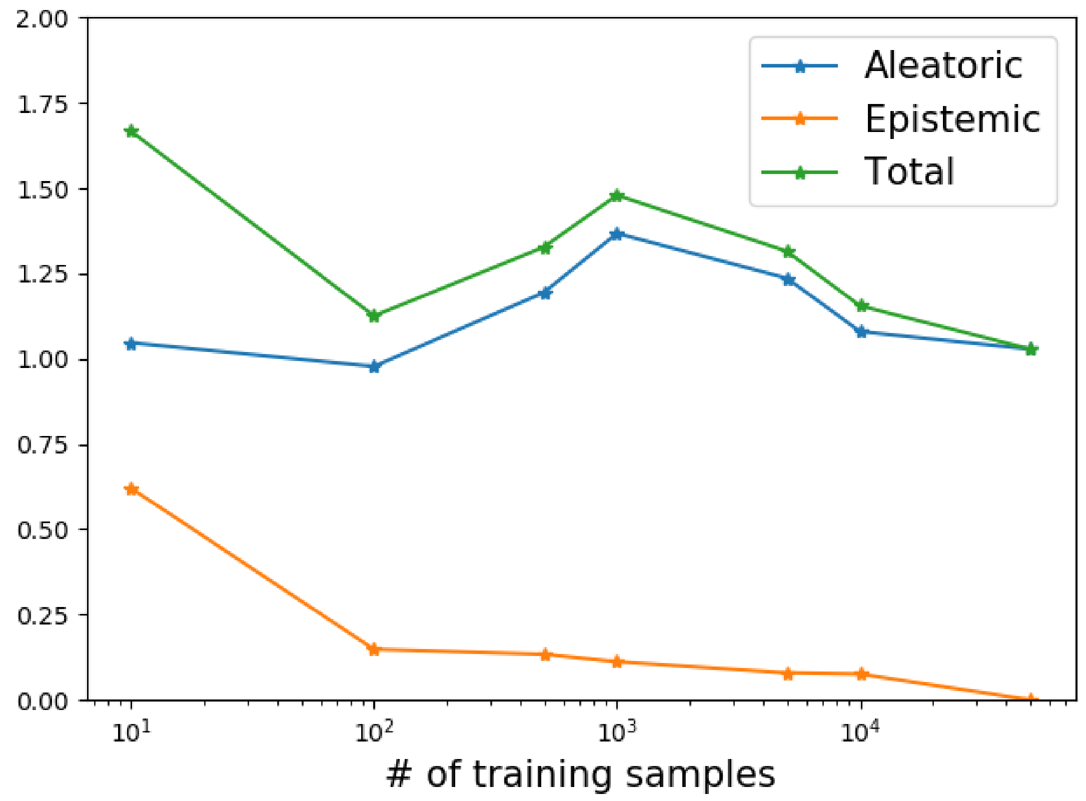
Bayesian inference

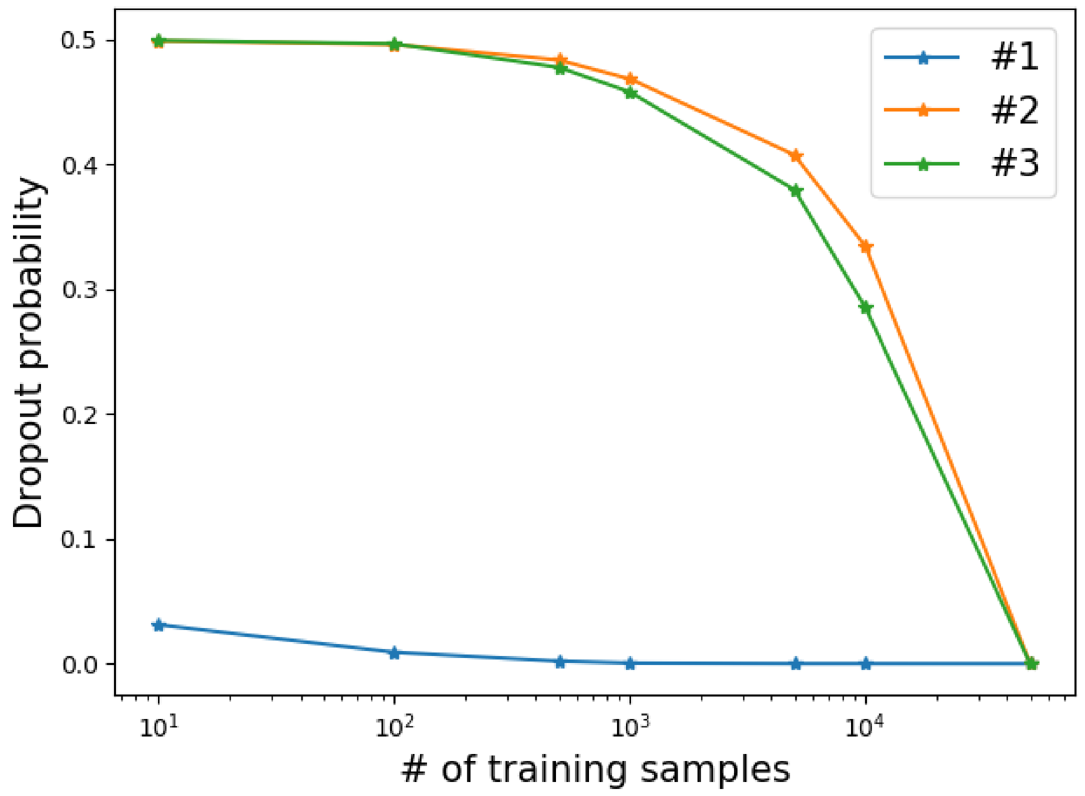


Toy model) $y = 2x + 8 + \epsilon, \quad \epsilon \sim \mathcal{N}(0,1)$

Toy model

1. The amount of training data increases

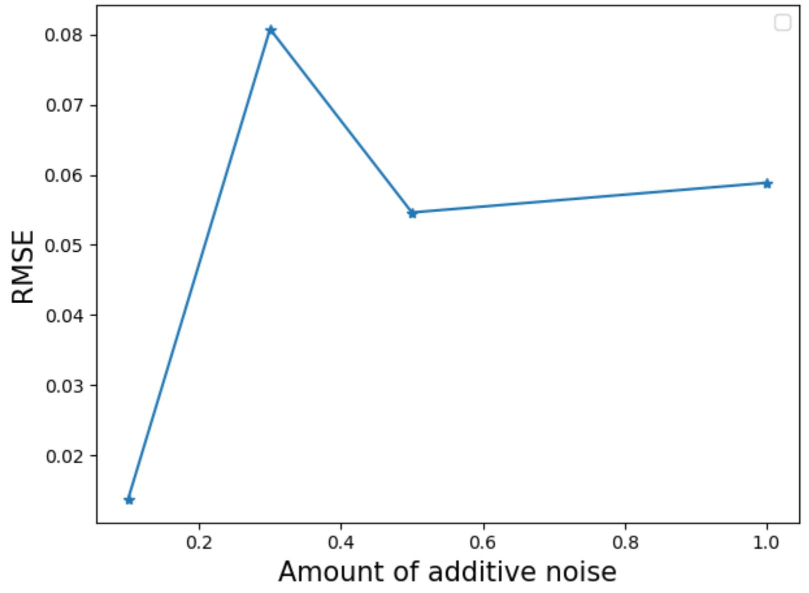


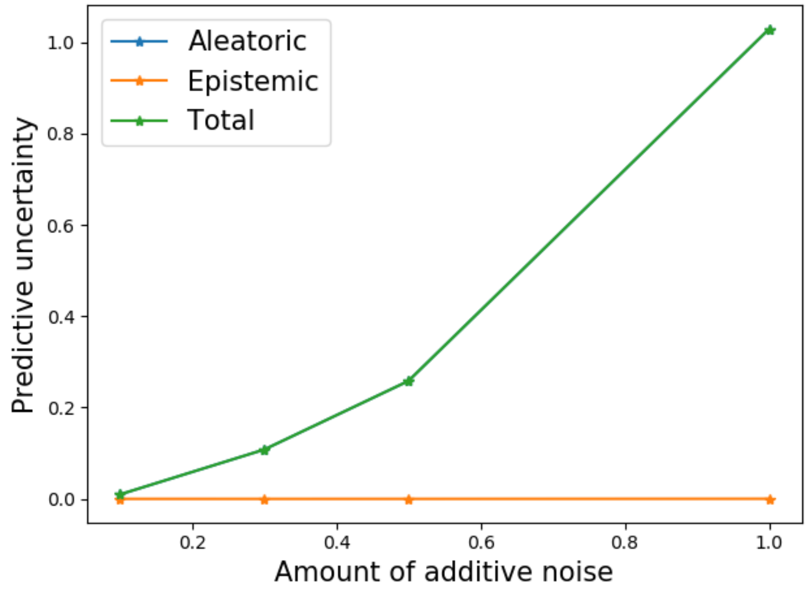


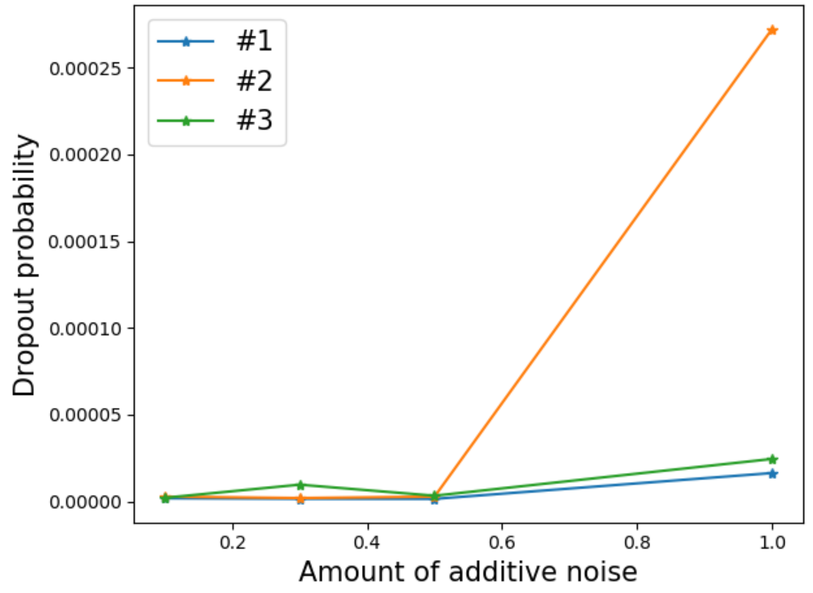
Toy model) $y = 2x + 8 + \epsilon, \quad \epsilon \sim \mathcal{N}(0,1)$

Toy model

2. The amount of additive noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$ increases







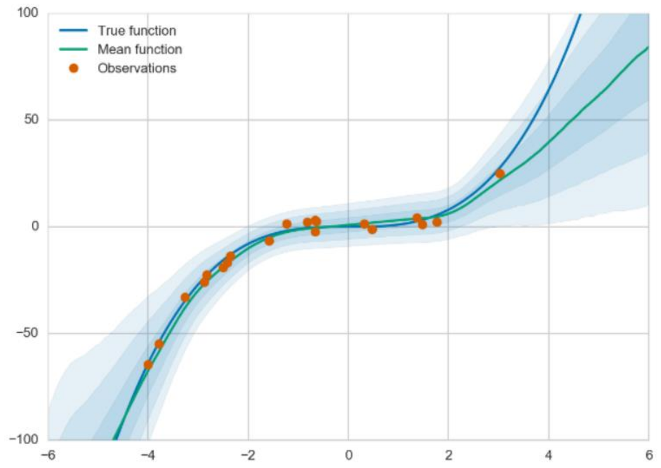
Toy model) $y = 2x + 8 + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2)$

Discussion

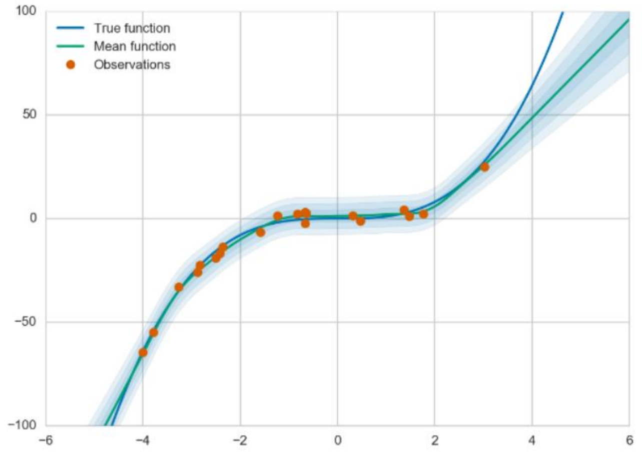
- Why did I use the Dropout network for Bayesian modeling?
  → Because… It is simple, easy to implement and cost-effective.

- Is the Dropout network good enough?
  → Cannot sure. There are two open questions for the Dropout network
    1) Is our choice of prior (for example $p(\boldsymbol{\omega}) = \mathcal{N}(0, \sigma^2 I; \boldsymbol{\omega})$) reasonable?
    2) Is using dropout for the re-parameterization trick reasonable?
    It is related to the quality of approximated posterior

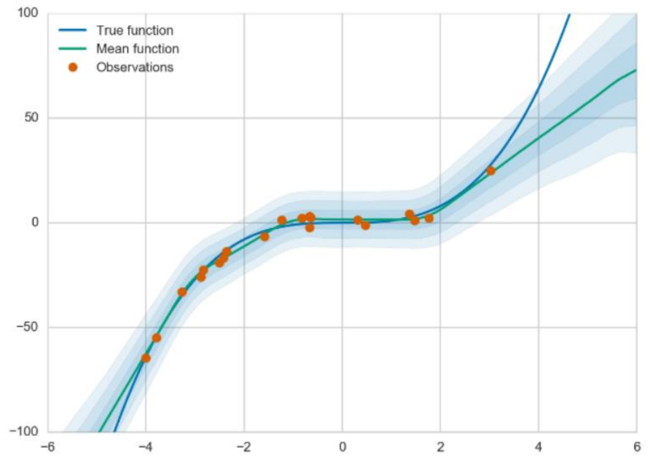
Discussion



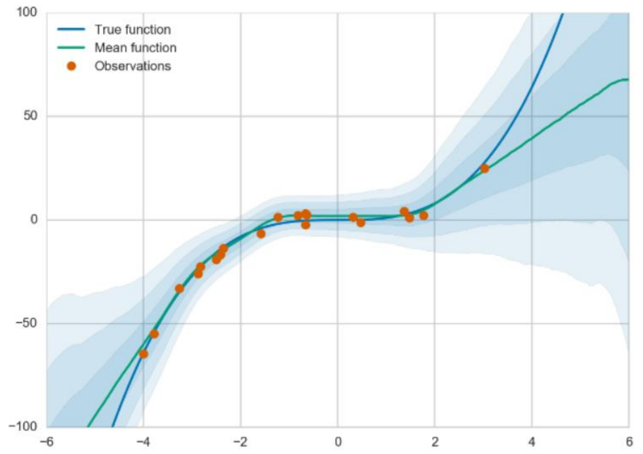
(a) Dropout $\pi = 0.5$



(b) Dropout learned π



(c) FFLU



(d) MNFG

Is the dropout network really a good posterior approximator?

Blue : true function

Green : predictive mean

Louizos, Christos, and Max Welling. "Multiplicative normalizing flows for variational bayesian neural networks." *arXiv preprint arXiv:1703.01961* (2017).

Discussion

- Why did I use the Dropout network for Bayesian modeling?
  → Because… It is simple, easy to implement and cost-effective.

- Is the Dropout network good enough?
  → Cannot sure. There are two open questions for the Dropout network
    1) Is our choice of prior (for example $p(\boldsymbol{\omega}) = \mathcal{N}(0, \sigma^2 I; \boldsymbol{\omega})$) reasonable?
    2) Is using dropout for the re-parameterization trick reasonable?
    It is related to the quality of approximated posterior

- Other Bayesian modeling methods are introduced in the previous slide.

Approximations in Bayesian modeling

Today's topic

1. **Variational inference: $p(\omega|X, Y) \approx q_\theta(\omega)$**
   - Fully Factorized Gaussian (FFG), also referred as mean-field approximation
   Blundell, Charles, et al. "Weight uncertainty in neural networks." *arXiv preprint arXiv:1505.05424* (2015).
   - Multiplicative Normalizing Flow (MNF)
   Louizos, Christos, and Max Welling. "Multiplicative normalizing flows for variational bayesian neural networks." *arXiv preprint arXiv:1703.01961* (2017).
   - Dropout network
   Gal, Yarin, and Zoubin Ghahramani. "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning." *international conference on machine learning*. 2016.
   - …

2. **Laplace approximation: pointwise estimation assisted with posterior curvature.**

3. **Markov chain Monte Carlo (MCMC): running Markov chains for Monte Carlo estimate of the posterior.**

Discussion

- Why did I use the Dropout network for Bayesian modeling?
  → Because… It is simple, easy to implement and cost-effective.

- Is the Dropout network good enough?
  → Cannot sure. There are two open questions for the Dropout network
    1) Is our choice of prior (for example $p(\boldsymbol{\omega}) = \mathcal{N}(0, \sigma^2 I; \boldsymbol{\omega})$) reasonable?
    2) Is using dropout for the re-parameterization trick reasonable?
    It is related to the quality of approximated posterior

- Other Bayesian modeling techniques able to adapt is introduced in the previous slide.

- Applications of Bayesian deep learning
  → Uncertainty-aware deep learning, for example, uncertainty-aware exploration and exploitation in reinforcement learning
  → Ideal Bayesian neural network can perfectly defense to the adversarial attack
  → …