Data Subsampling for Bayesian Neural Networks

Penalty Bayesian Neural Networks

Eiji Kawasaki
01/12/2023
CEA/DRT/LIST

AISSAI
CNRS AISSAI IN2P3

Acknowledgements

CEA

DRT LIST - François TERRIER, Cédric AULIAC, Eric BARAT, Thomas DAUTREMER, Han WANG, Fabio-Alejandro ARNEZ YAGUALCA, Ansgar RADERMACHER, Victor BERGER, Aurélien BENOIT-LEVY, Thomas DALGATY

Students, PhDs and Postdoctoral researchers - Elouan ARGOUARC'H, Fedor GONCHAROV, Lawrence ADU-GYAMFI, Dimitrios TZIVRAILIS, Arsène FERRIERE

DES - Geoffrey DANIEL, Guillaume DAMBLIN, Clément GAUCHY

CNRS

LPTMS Alberto ROSSO, **LPMMC** Markus HOLZMANN

Data Subsampling for Bayesian Neural Networks
arXiv:2210.09141
Eiji KAWASAKI & Markus HOLZMANN

Confiance ai

Bayesian Posterior Predictive Distribution

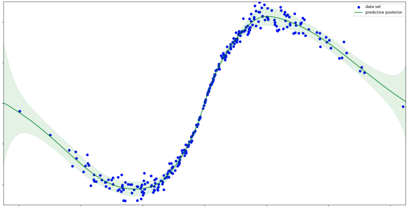
We consider the **Neural Network parameters** θ as random variables. BNN quantify the **epistemic uncertainty** coming from the posterior probability distribution $p(\theta|\mathcal{D})$.

Notations

$\mathcal{D} = (y_i, x_i)_{i=1}^{L}$	Training data set
x	Input
y	Output
$p_\theta(y\|x)$	Likelihood parameterized by a Neural Network

Taking into account this uncertainty while making a prediction means to marginalize over all possible parameters values θ.

$$p(y|x, \mathcal{D}) = \int d\theta \; p_\theta(y|x) p(\theta|\mathcal{D})$$



Works on generative modelling as well: $p(x|\mathcal{D}) = \int d\theta \; p_\theta(x) p(\theta|\mathcal{D})$

Posterior Predictive Monte Carlo Estimation

The predictive distribution $p(y|x,\mathcal{D})$ can be approximated using a Monte Carlo estimate,

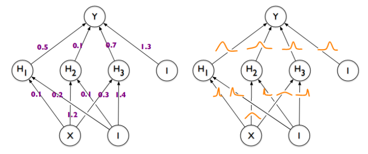
$$p(y|x,\mathcal{D}) = \int d\theta \; p_\theta(y|x)p(\theta|\mathcal{D}) = \mathbb{E}_{p(\theta|\mathcal{D})}[p_\theta(y|x)]$$

$$\overset{MC}{\simeq} \frac{1}{N}\sum_{i=1}^{N} p_{\theta^{(i)}}(y|x) \qquad \text{with } \theta^{(i)} \sim p(\theta|\mathcal{D})$$

turning the Uncertainty Quantification into a problem of sampling $p(\theta|\mathcal{D})$.

Bayes By Backprop [2]

Blundell & alt approximate the posterior distribution $p(\theta|\mathcal{D})$ as a diagonal Gaussian distribution (Variational Inference).



Bayes' theorem and the intractable normalising factor

Using Bayes' theorem, we write the posterior distribution $p(\theta|\mathcal{D})$ as

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D},\theta)}{p(\mathcal{D})} = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}$$
$$= \frac{\prod_{i=1}^{L} p_\theta(y_i|x_i)p(\theta)}{p(\mathcal{D})} = \frac{e^{-\mathcal{L}_\mathcal{D}(\theta)}}{p(\mathcal{D})}$$

The normalising factor $p(\mathcal{D})$ is unknown and cannot be computed. We thus need to **sample a distribution known up to a constant**. The loss function $\mathcal{L}_\mathcal{D}(\theta)$ is defined as

$$\mathcal{L}_\mathcal{D}(\theta) = -\sum_{i=1}^{L} \log p_\theta(y_i|x_i) - \log p(\theta)$$

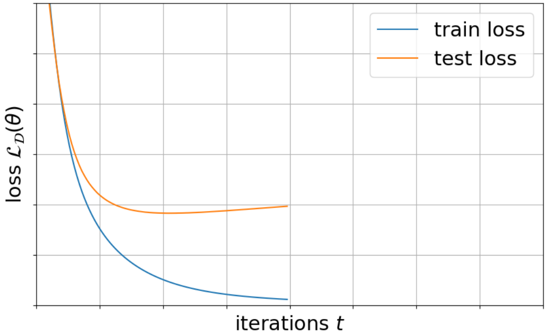
Regression loss example

$$\mathcal{L}_\mathcal{D}^{\text{Reg}}(\theta) = \sum_{i=1}^{L} (y_i - f_\theta(x_i))^2 + \lambda \sum_j |\theta_j|^2 + \text{cst}$$

where $f_\theta(x)$ is a Neural Network.

$\mathcal{L}_{\mathcal{D}}(\theta)$ Minimization

MNIST Handwritten Digit Classification

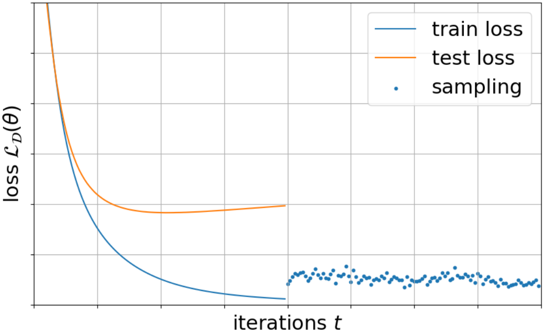


Using a Gradient Descent Algorithm $\theta_{t+1} = \theta_t - \eta \boldsymbol{\nabla}_\theta \mathcal{L}_{\mathcal{D}}(\theta_t)$

we expect a **prediction error** e.g. $\mathbb{E}\left[(f_{\theta_t}(x) - y)^2\right] \geq 0$.

$p(\theta|\mathcal{D})$ Sampling

MNIST Handwritten Digit Classification



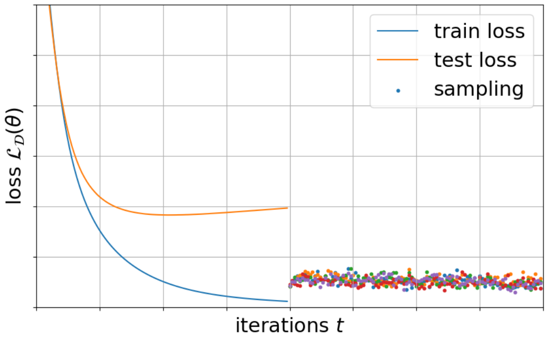
BNNs replace a **point estimate** of $p(\theta|\mathcal{D})$ by samples $\theta^{(t)} \sim p(\theta|\mathcal{D}) = \frac{e^{-\mathcal{L}_{\mathcal{D}}(\theta)}}{p(\mathcal{D})}$.

The loss $\mathcal{L}_{\mathcal{D}}(\theta)$ is not minimized: its expected value is determined by the entropy.

$$\langle \mathcal{L}_{\mathcal{D}}(\theta) \rangle_{p(\theta|\mathcal{D})} = -\log p(\mathcal{D}) - \int d\theta \; p(\theta|\mathcal{D}) \log p(\theta|\mathcal{D})$$

$p(\theta|\mathcal{D})$ Sampling Ergodicity

MNIST Handwritten Digit Classification

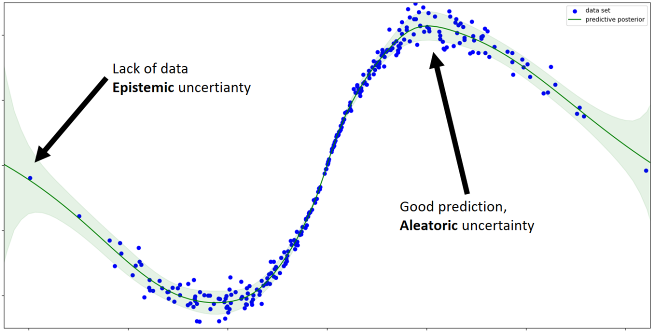


$\mathcal{L}_{\mathcal{D}}(\theta)$ has many **local minima**, that often correspond to similar functions $p_\theta(y|x)$.

Mode exploration vs Mode exploitation $\rightarrow$ Deep Ensembles, SGD, Multi-SWAG [10]

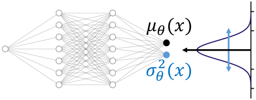
Bayesian Neural Network for Regression

$$p(y|x,\mathcal{D}) \simeq \frac{1}{N}\sum_{i=1}^{N} p_{\theta^{(i)}}(y|x) \qquad \text{with } \theta^{(i)} \sim p(\theta|\mathcal{D})$$



Mixture Density Network:

$$p_\theta(y|x) = \mathcal{N}(y|\mu_\theta(x), \sigma^2_\theta(x))$$



BNN State Of The Art [5]

How can we obtain samples $\theta \sim p(\theta|\mathcal{D})$ where $p(\theta|\mathcal{D}) = e^{-\mathcal{L}_{\mathcal{D}}(\theta)}/p(\mathcal{D})$?

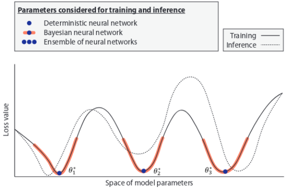
Fast Biased Estimates

Deep Ensemble methods

Monte Carlo Dropout

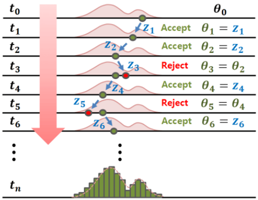
Variational Inference

Laplace Approximation



Slow Unbiased Estimate - Gold Standard

We design a **Markov Chain** that asymptotically reaches a unique stationary distribution $p(\theta|\mathcal{D})$ [6].



Markov Chain Monte Carlo

We design a transition probability $T(\theta_t \to \theta')$:

$$T(\theta_t \to \theta') = q(\theta'|\theta_t)\mathcal{A}(\theta', \theta_t)$$

where $q(\theta'|\theta_t)$ is a **proposal distribution** and $\mathcal{A}(\theta', \theta_t)$ is an **acceptance probability**.

In order to sample the distribution $p(\theta_t|\mathcal{D})$, a sufficient but not necessary condition is the **Detailed Balance**.

$$p(\theta_t|\mathcal{D})\,T(\theta_t \to \theta') = p(\theta'|\mathcal{D})\,T(\theta' \to \theta_t)$$

It follows that satisfying the Detailed Balance means that

$$\begin{aligned}\mathcal{A}(\theta', \theta_t) &= \min\left(1, \frac{p(\theta'|\mathcal{D})}{p(\theta_t|\mathcal{D})}\frac{q(\theta_t|\theta')}{q(\theta'|\theta_t)}\right)\\ &= \min\left(1, \frac{e^{-\mathcal{L}_{\mathcal{D}}(\theta')}}{e^{-\mathcal{L}_{\mathcal{D}}(\theta_t)}}\frac{q(\theta_t|\theta')}{q(\theta'|\theta_t)}\right)\end{aligned}$$

MCMC obtains samples $\theta \sim p(\theta|\mathcal{D})$ without computing the normalizing constant $p(\mathcal{D})$.

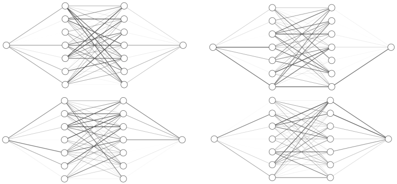
Metropolis-Hastings Random Walk algorithm

We compute the predictive posterior distribution that takes into account the aleatoric and epistemic uncertainties.

$$p(y|x,\mathcal{D}) \simeq \frac{1}{N}\sum_{i=1}^{N} p_{\theta^{(i)}}(y|x)$$

$$\theta^{(i)} \sim p(\theta|\mathcal{D})$$

$\theta^{(0)}, \theta^{(1)}, \dots, \theta^{(N-1)}$ correspond to different parameters of the same neural network model.



Algorithm 1 Random Walk algorithm

$t \leftarrow 0$
$\theta_t \leftarrow \theta_0$
for N **do**
  $\epsilon_t \leftarrow$ sample $\mathcal{N}(0,1)$
  $\theta' \leftarrow \theta_t + \eta\epsilon_t$
  $\Delta(\theta',\theta_t) \leftarrow \mathcal{L}_{\mathcal{D}}(\theta') - \mathcal{L}_{\mathcal{D}}(\theta_t)$
  $\mathcal{A}(\theta',\theta_t) \leftarrow \min\left(1, e^{-\Delta(\theta',\theta_t)}\right)$
  $u \leftarrow$ sample $\mathcal{U}(0,1)$
  if $u \leq \mathcal{A}(\theta',\theta_t)$ **then**
    $\theta_{t+1} \leftarrow \theta'$
  else
    $\theta_{t+1} \leftarrow \theta_t$
  end if
  $t \leftarrow t+1$
end for

Challenges in MCMC for BNN [7]

Besides common BNN limitations (e.g. weight symmetries and prior specification) MCMC for BNN has specific challenges:

Size of θ: model size scalability is challenging. It is difficult to sample a high-dimensional distribution $p(\theta|\mathcal{D})$.

Size of $\mathcal{D}$: data size scalability is challenging. Large data set loss computation $\mathcal{L}_{\mathcal{D}}(\theta)$ slows down the MCMC sampling.

Design a sampling methods suited for Deep Learning

Differentiable model: Neural Network have differentiable losses.

Data Subsampling: it is possible to create data mini-batches.

Over-parametrized: partial semi-stochastic BNN [8].

Gradient Based Monte Carlo Proposal

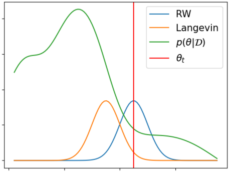
We Taylor expand the loss

$$\mathcal{L}_{\mathcal{D}}(\theta_t) \approx \mathcal{L}_{\mathcal{D}}(\theta') + (\theta_t - \theta') \cdot \nabla_\theta \mathcal{L}_{\mathcal{D}}(\theta')$$

$$\mathcal{A}(\theta', \theta_t) \approx \min\left(1, \frac{q(\theta_t|\theta')}{q(\theta'|\theta_t)} e^{-(\theta_t - \theta')\cdot(\nabla_\theta \mathcal{L}_{\mathcal{D}}(\theta') + \nabla_\theta \mathcal{L}_{\mathcal{D}}(\theta_t))/2}\right)$$

We choose a Gaussian proposal distribution to locally approximate the posterior distribution. Maximizing $A(\theta', \theta_t)$ leads to

$$q(\theta'|\theta_t) = \mathcal{N}(\theta'; \theta_t - \eta \nabla_\theta \mathcal{L}_{\mathcal{D}}(\theta_t), 2\eta)$$



Sampling a new state θ' from the proposal distribution $q(\theta'|\theta_t)$ corresponds exactly to drawing a centered reduced normal random variable ϵ_t, and computing

$$\theta' = \theta_t - \eta \nabla_{\theta_t} \mathcal{L}(\theta_t) + \sqrt{2\eta}\epsilon_t$$

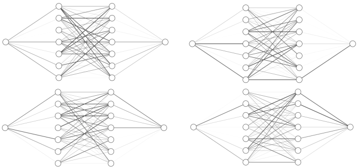
Metropolis-Adjusted Langevin Algorithm

We compute the predictive posterior distribution that takes into account the aleatoric and epistemic uncertainties.

$$p(y|x, \mathcal{D}) \simeq \frac{1}{N} \sum_{i=1}^{N} p_{\theta^{(i)}}(y|x)$$

$$\theta^{(i)} \sim p(\theta|\mathcal{D})$$

$\theta^{(0)}, \theta^{(1)}, \dots, \theta^{(N-1)}$ correspond to different weights of the same neural network model.



Algorithm 2 MALA
Metropolis-Adjusted Langevin Algorithm

$t \leftarrow 0$
$\theta_t \leftarrow \theta_0$
for t in N **do**
    $\epsilon_t \sim \mathcal{N}(0, 1)$
    $\theta' \leftarrow \theta_t - \eta \boldsymbol{\nabla}_\theta \mathcal{L}_\mathcal{D}(\theta_t) + \sqrt{2\eta}\epsilon_t$
    $\Delta(\theta', \theta_t) \leftarrow \mathcal{L}_\mathcal{D}(\theta') - \mathcal{L}_\mathcal{D}(\theta_t)$
    $\mathcal{A}(\theta', \theta_t) \leftarrow \min\left(1, \frac{q(\theta|\theta')}{q(\theta'|\theta)} e^{-\Delta(\theta', \theta_t)}\right)$
    $u \leftarrow$ sample $\mathcal{U}(0, 1)$
    if $u \leq A(\theta', \theta_t)$ **then**
        $\theta_{t+1} \leftarrow \theta'$
    else
        $\theta_{t+1} \leftarrow \theta_t$
    end if
end for

Requires the computation of the loss $\mathcal{L}_\mathcal{D}(\theta)$ over the full dataset.

Training Data Subsampling

Our goal is to design a data subsampling strategy. For a given mini-batch size L^{MB}, the expected loss writes

$$\mathcal{L}_{MB}(\theta) = -\log p(\theta) - L^{MB}\mathbb{E}\left[\log p_\theta(y|x)\right]$$

Random walk Metropolis-Hastings acceptance

$$A(\theta', \theta_t) = \min\left(1, e^{-\Delta(\theta', \theta_t)}\right)$$

$$\Delta(\theta', \theta_t) = \mathcal{L}_{MB}(\theta') - \mathcal{L}_{MB}(\theta_t)$$

Instead of the expected loss difference $\Delta(\theta', \theta_t)$, we observe a random variable which we assume as normally distributed

$$\delta(\theta', \theta_t) \sim \mathcal{N}(\Delta(\theta', \theta_t), \sigma^2(\theta', \theta_t))$$

Noisy loss difference

We define the observed loss difference as an empirical average over M mini-batches.

$$\delta(\theta', \theta_t) = \frac{1}{M}\sum_{j=1}^{M}\left(\mathcal{L}^j_{MB}(\theta') - \mathcal{L}^j_{MB}(\theta_t)\right)$$

Noise Penalty Method

In the Journal of Physical Chemistry (1999) [4], Ceperley and Dewing have generalized the Metropolis-Hastings random walk algorithm to the situation where the loss is noisy and can only be estimated.

Noise penalty acceptance

$$A(\delta, \theta', \theta_t) = \min\left(1, e^{-\delta(\theta', \theta_t) - \sigma^2(\theta', \theta_t)/2}\right)$$

Using the noise penalty $e^{-\sigma^2(\theta', \theta_t)/2}$ (that suppresses the acceptance probability) one can show that **detailed balance is satisfied on average**. $\sigma^2(\theta', \theta_t)$ is unknown and needs to be estimated.

$\sigma^2(\theta', \theta_t)$ chi-squared estimator

$$\chi^2(\theta', \theta_t) = \frac{1}{M(M-1)} \sum_{j=1}^{M} \left(\mathcal{L}^j_{MB}(\theta') - \mathcal{L}^j_{MB}(\theta_t) - \delta(\theta', \theta_t)\right)^2$$

Penalty BNN Numerical Results

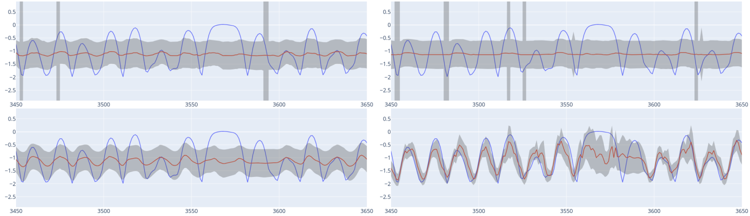
arXiv:2210.09141

Tempered BNN

$$-\log p(\theta) - \frac{L^{MB}}{L}\sum_{i=1}^{L}\log p(y_i|x_i,\theta)$$

Batched BNN

$$A(\delta,\theta',\theta_t) = \min(1, e^{-\delta(\theta',\theta_t)})$$



Stochastic Gradient Langevin Dynamics

$$\theta_{t+1} = \theta_t - \eta_t \nabla_\theta \mathcal{L}_{MB}(\theta_t) + \sqrt{2\eta_t}\epsilon_t$$

Penalty BNN

$$A(\delta,\theta',\theta_t) = \min\left(1, e^{-\delta(\theta',\theta_t)-\chi^2(\theta',\theta_t)/2}\right)$$

The literature of SGLD [9] and Barker acceptance test [1] commonly sample a biased posterior and then try to control this bias, e.g. reducing it below a threshold [3].

Conclusion

A noisy estimate of the loss introduces a **bias in BNN's posterior sampling** if not taken into account.

Removing this bias requires a **noise penalty** that corresponds to the variance of the noisy loss difference.

Algorithm 3 Random Walk PBNN Algorithm

$t \leftarrow 0$
$\theta_t \leftarrow \theta_0$
for N **do**
    $\delta_t \leftarrow$ sample $\mathcal{N}(0,1)$
    $\theta' \leftarrow \theta_t + \eta\delta_t$
    $\delta(\theta',\theta_t) \leftarrow \frac{1}{M}\sum_{j=1}^{M}\left(\mathcal{L}^j_{MB}(\theta') - \mathcal{L}^j_{MB}(\theta_t)\right)$
    $\chi^2(\theta',\theta_t) \leftarrow \frac{1}{M(M-1)}\sum_{j=1}^{M}\left(\mathcal{L}_{\mathcal{D}_j}(\theta') - \mathcal{L}_{\mathcal{D}_j}(\theta_t) - \delta(\theta',\theta_t)\right)^2$
    $\mathcal{A}(\theta',\theta_t) \leftarrow \min\left(1, e^{-\delta(\theta',\theta_t)-\chi^2(\theta',\theta_t)/2}\right)$
    $u \leftarrow$ sample $\mathcal{U}(0,1)$
    if $u \leq \mathcal{A}(\theta',\theta_t)$ **then**
        $\theta_{t+1} \leftarrow \theta'$
    else
        $\theta_{t+1} \leftarrow \theta_t$
    end if
    $t \leftarrow t+1$
end for

References (1/2)

[1] Rémi Bardenet, Arnaud Doucet, and Chris Holmes. **"On Markov chain Monte Carlo methods for tall data".** In: *Journal of Machine Learning Research* 18.47 (2017), pp. 1–43. URL: http://jmlr.org/papers/v18/15-205.html.

[2] Charles Blundell et al. **"Weight Uncertainty in Neural Networks".** In: *arXiv:1505.05424 [cs, stat]* (May 2015). arXiv: 1505.05424. URL: http://arxiv.org/abs/1505.05424.

[3] Nicolas Brosse, Alain Durmus, and Eric Moulines. **"The promises and pitfalls of Stochastic Gradient Langevin Dynamics".** In: *arXiv:1811.10072 [cs, stat]* (Nov. 2018). arXiv: 1811.10072. URL: http://arxiv.org/abs/1811.10072.

[4] D. M. Ceperley and M. Dewing. **"The penalty method for random walks with uncertain energies".** en. In: *The Journal of Chemical Physics* 110.20 (May 1999), pp. 9812–9820. ISSN: 0021-9606, 1089-7690. DOI: 10.1063/1.478034. URL: http://aip.scitation.org/doi/10.1063/1.478034.

[5] Jakob Gawlikowski et al. **"A Survey of Uncertainty in Deep Neural Networks".** In: *arXiv:2107.03342 [cs, stat]* (July 2021). arXiv: 2107.03342. URL: http://arxiv.org/abs/2107.03342.

References (2/2)

[6] Radford M. Neal. **Bayesian Learning for Neural Networks.** en. Ed. by P. Bickel et al. Vol. 118. Lecture Notes in Statistics. New York, NY: Springer New York, 1996. ISBN: 978-0-387-94724-2 978-1-4612-0745-0. DOI: 10.1007/978-1-4612-0745-0. URL: http://link.springer.com/10.1007/978-1-4612-0745-0.

[7] Theodore Papamarkou et al. **"Challenges in Markov chain Monte Carlo for Bayesian neural networks".** In: *arXiv:1910.06539 [cs, stat]* (Oct. 2021). arXiv: 1910.06539. URL: http://arxiv.org/abs/1910.06539.

[8] Mrinank Sharma et al. **Do Bayesian Neural Networks Need To Be Fully Stochastic?** Number: arXiv:2211.06291 arXiv:2211.06291 [cs, stat]. Nov. 2022. URL: http://arxiv.org/abs/2211.06291.

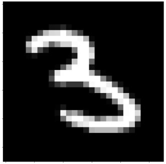
[9] Max Welling and Yee Whye Teh. **"Bayesian Learning via Stochastic Gradient Langevin Dynamics".** en. In: (), p. 8.

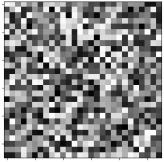
[10] Andrew Gordon Wilson and Pavel Izmailov. **"Bayesian Deep Learning and a Probabilistic Perspective of Generalization".** In: *arXiv:2002.08791 [cs, stat]* (Apr. 2020). arXiv: 2002.08791. URL: http://arxiv.org/abs/2002.08791.

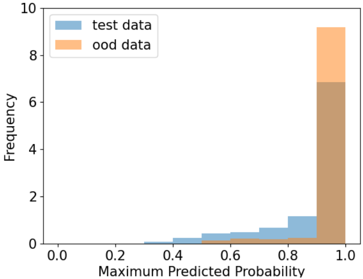
Thank you!

MNIST Classification OOD example

After training a classifier on the MNIST database (handwritten digits), we create a histogram of the predicted probability label for two data sets: one **test data set** and one **out-of-distribution data set** containing pure noise.

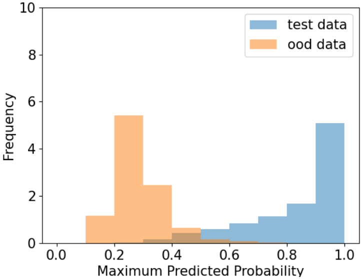






$$\theta \leftarrow \theta - \eta \boldsymbol{\nabla}_\theta \mathcal{L}_\mathcal{D}(\theta)$$

prediction: $p_\theta(y|x)$



$$\theta_{t+1} \leftarrow \theta_t - \eta \boldsymbol{\nabla}_\theta \mathcal{L}_\mathcal{D}(\theta_t) + \sqrt{2\eta}\delta_t$$

prediction: $p(y|x,\mathcal{D}) \simeq \frac{1}{N}\sum_{i=1}^{N} p_{\theta^{(i)}}(y|x)$