Deep Learning with Bayesian Principles

Mohammad Emtiyaz Khan

RIKEN Center for AI Project, Tokyo

http://emtiyaz.github.io

With a significant help from

Roman Bachmann (RIKEN-AIP)

Xiangming Meng (RIKEN-AIP)

The Goal of My Research

"To discover the fundamental principles of learning from data and use them to develop algorithms that can learn like living beings."

Human Learning at the age of 6 months.

Human Learning at the age of 6 months.

Human Learning at the age of 6 months.

Converged at the age of 12 months

Converged at the age of 12 months

Converged at the age of 12 months

Transfer skills

at the age of 14 months

Transfer skills

at the age of 14 months

Transfer skills

at the age of 14 months

Human learning $\neq$ **Deep learning**

Life-long learning from small chunks of data in a non-stationary world

Bulk learning from a large amount of data in a stationary world

Parisi, German I., et al. "Continual lifelong learning with neural networks: A review." *Neural Networks* (2019)

Human learning $\neq$ **Deep learning**

Life-long learning from small chunks of data in a non-stationary world

Bulk learning from a large amount of data in a stationary world

My current research focuses on reducing this gap!

Parisi, German I., et al. "Continual lifelong learning with neural networks: A review." *Neural Networks* (2019)

Bayesian

~~Human~~ learning $\neq$ **Deep learning**

Life-long learning from small chunks of data in a non-stationary world

Bulk learning from a large amount of data in a stationary world

My current research focuses on reducing this gap!

Parisi, German I., et al. "Continual lifelong learning with neural networks: A review." *Neural Networks* (2019)

Bayesian
~~Human~~ learning $\neq$ **Deep learning**

Life-long learning from small chunks of data in a non-stationary world

Bulk learning from a large amount of data in a stationary world

My current research focuses on reducing this gap!

Parisi, German I., et al. "Continual lifelong learning with neural networks: A review." *Neural Networks* (2019)
Friston, K. "The free-energy principle: a unified brain theory?." *Nature reviews neuroscience* (2010)
Geisler, W. S., and Randy L. D. "Bayesian natural selection and the evolution of perceptual systems." *Philosophical Transactions of the Royal Society of London. Biological Sciences* (2002)

Bayesian learning

Bayesian models

(GPs, BayesNets, PGMs,)

Bayesian inference

(Bayes rule)

Deep learning

Deep models

(MLP, CNN, RNN etc.)

Stochastic training

(SGD, RMSprop, Adam)

Bayesian learning

Bayesian models

(GPs, BayesNets, PGMs,)

Bayesian inference

(Bayes rule)

Deep learning

Deep models

(MLP, CNN, RNN etc.)

Stochastic training

(SGD, RMSprop, Adam)

	Bayes	DL
Can handle large data and complex models?	✗	✓
Scalable training?	✗	✓
Can estimate uncertainty?	✓	✗
Can perform sequential / active /online / incremental learning?	✓	✗

Bringing the two together

To combine their complimentary strengths to solve challenging learning problems

Deep Learning with Bayesian Principles

Deep Learning with Bayesian Principles

- Bayesian principles as a general principle
  - To design/improve/generalize learning-algorithms
  - By computing “posterior approximations”

Deep Learning with Bayesian Principles

- Bayesian principles as a general principle
  - To design/improve/generalize learning-algorithms
  - By computing "posterior approximations"
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc

Deep Learning with Bayesian Principles

- Bayesian principles as a general principle
  - To design/improve/generalize learning-algorithms
  - By computing "posterior approximations"
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and life-long learning

Deep Learning with Bayesian Principles

- Bayesian principles as a general principle
  - To design/improve/generalize learning-algorithms
  - By computing “posterior approximations”
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and life-long learning
- Impact: Everything with one common principle.

Is this different from Bayesian Deep Learning?

Scope of the Tutorial

- Audience: Deep learners and Bayesians
- Goal: To bring the two together
- This tutorial is not about
  - Bayesian deep-learning methods
  - Classical Bayesian inference methods
  - Approximate Bayesian Inference
  - Uncertainty estimation
  - Generative Models, VAE, etc.
  - Gaussian processes and NN architectures

Disclaimer

- I might not have time to discuss many important/relevant works
  - If you think I should have included some of those, please send me email and I will try to include it the next time
- The content of the tutorial is based on my own biased opinion (and expertise)
  - A lot of it is based on my own work (about 40% or so)

Deep Learning
vs
Bayesian Learning

Deep Learning (DL)

Frequentist: Empirical Risk Minimization (ERM) or Maximum Likelihood Principle, etc.

$$\min_{\theta} \ \ell(\mathcal{D}, \theta)$$

Loss

Data

Model Params

Deep Learning (DL)

Frequentist: Empirical Risk Minimization (ERM) or Maximum Likelihood Principle, etc.

$$\min_{\theta} \ \ell(\mathcal{D}, \theta) \ = \sum_{i=1}^{N} [y_i - f_\theta(x_i)]^2 + \gamma \theta^T \theta$$

Loss

Data

Model Params

Deep Network

Deep Learning (DL)

Frequentist: Empirical Risk Minimization (ERM) or Maximum Likelihood Principle, etc.

$$\min_{\theta} \ \ell(\mathcal{D}, \theta) \quad = \sum_{i=1}^{N} [y_i - f_\theta(x_i)]^2 + \gamma \theta^T \theta$$

Loss

Data

Model Params

Deep Network

DL Algorithm: $\theta \leftarrow \theta - \rho H_\theta^{-1} \nabla_\theta \ell(\theta)$

Deep Learning (DL)

Frequentist: Empirical Risk Minimization (ERM) or Maximum Likelihood Principle, etc.

$$\min_{\theta} \ \ell(\mathcal{D}, \theta) = \sum_{i=1}^{N} [y_i - f_\theta(x_i)]^2 + \gamma \theta^T \theta$$

Loss

Data

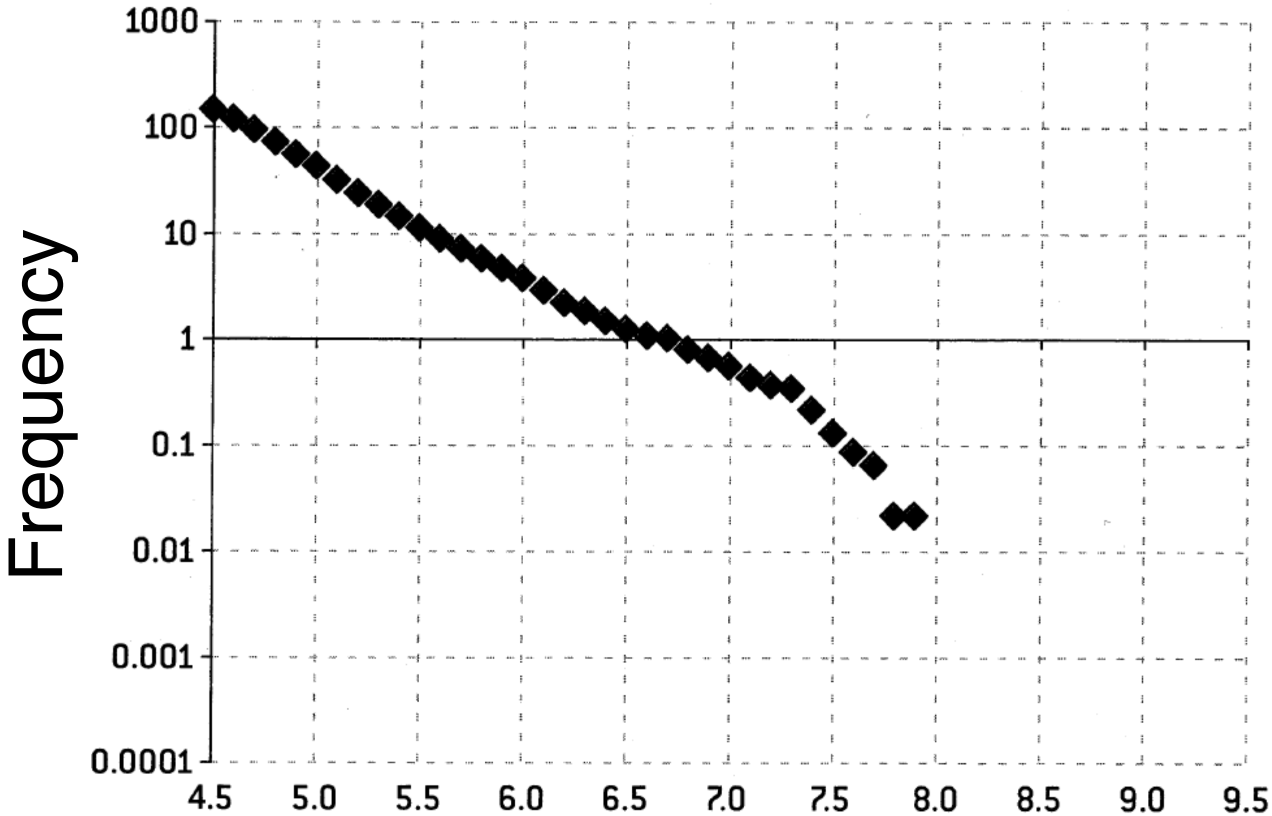
Model Params

Deep Network

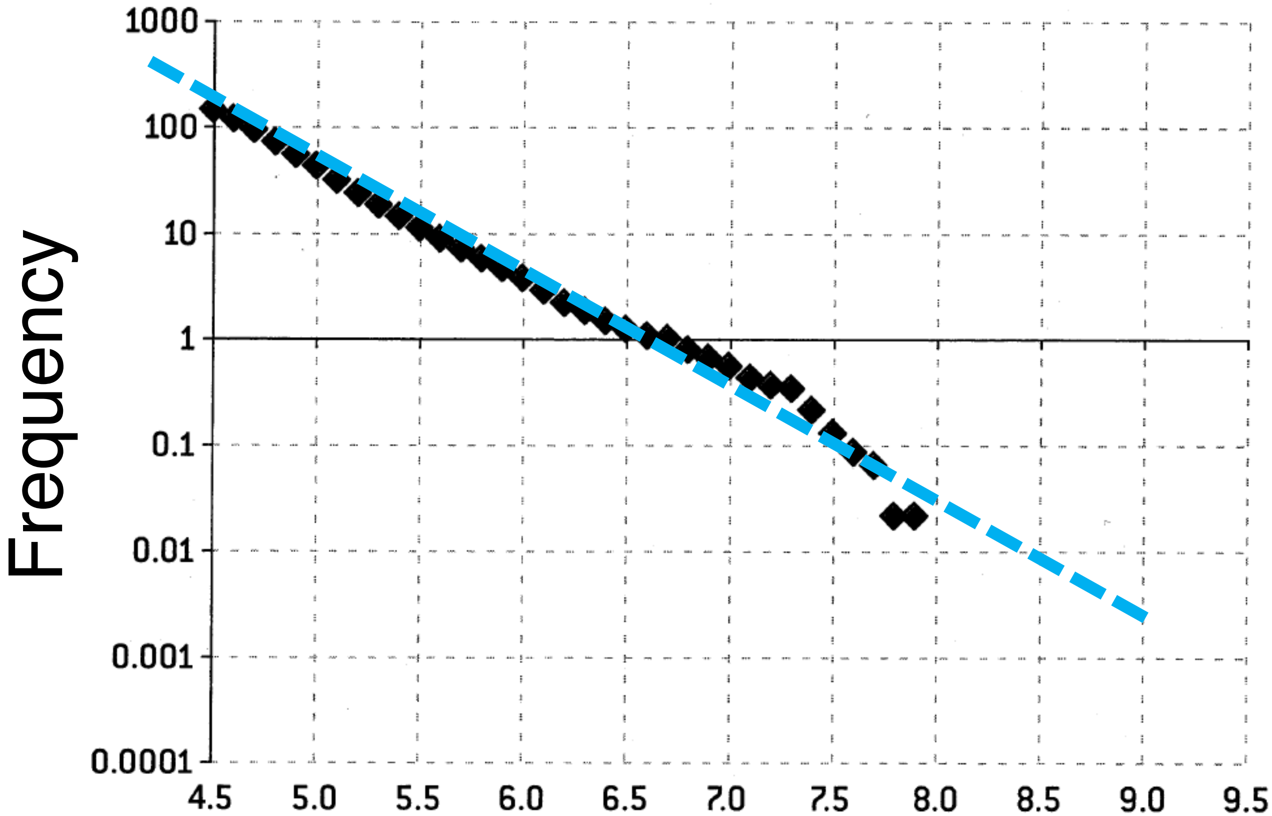
DL Algorithm: $\theta \leftarrow \theta - \rho H_\theta^{-1} \nabla_\theta \ell(\theta)$

Scales well to large data and complex model, and very good performance in practice.

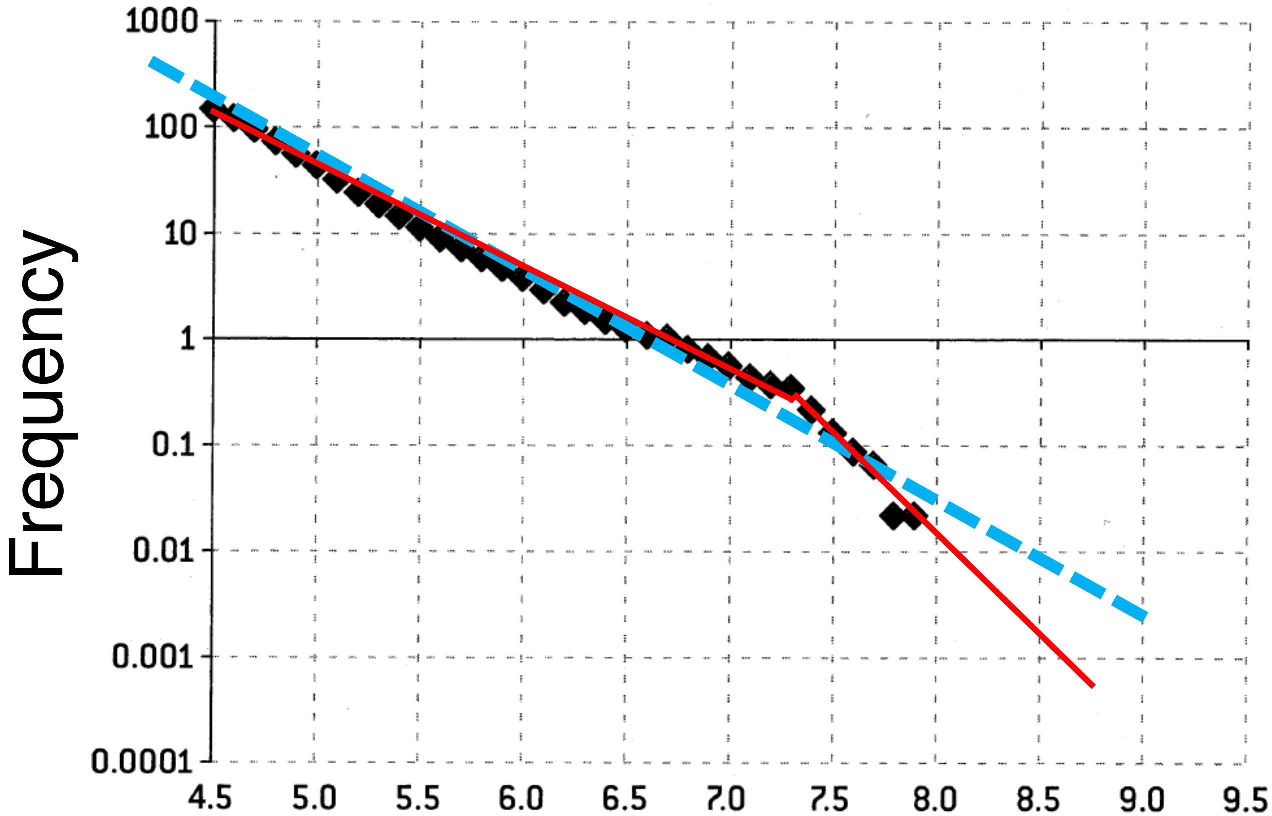
Example: Which is a Better Fit?



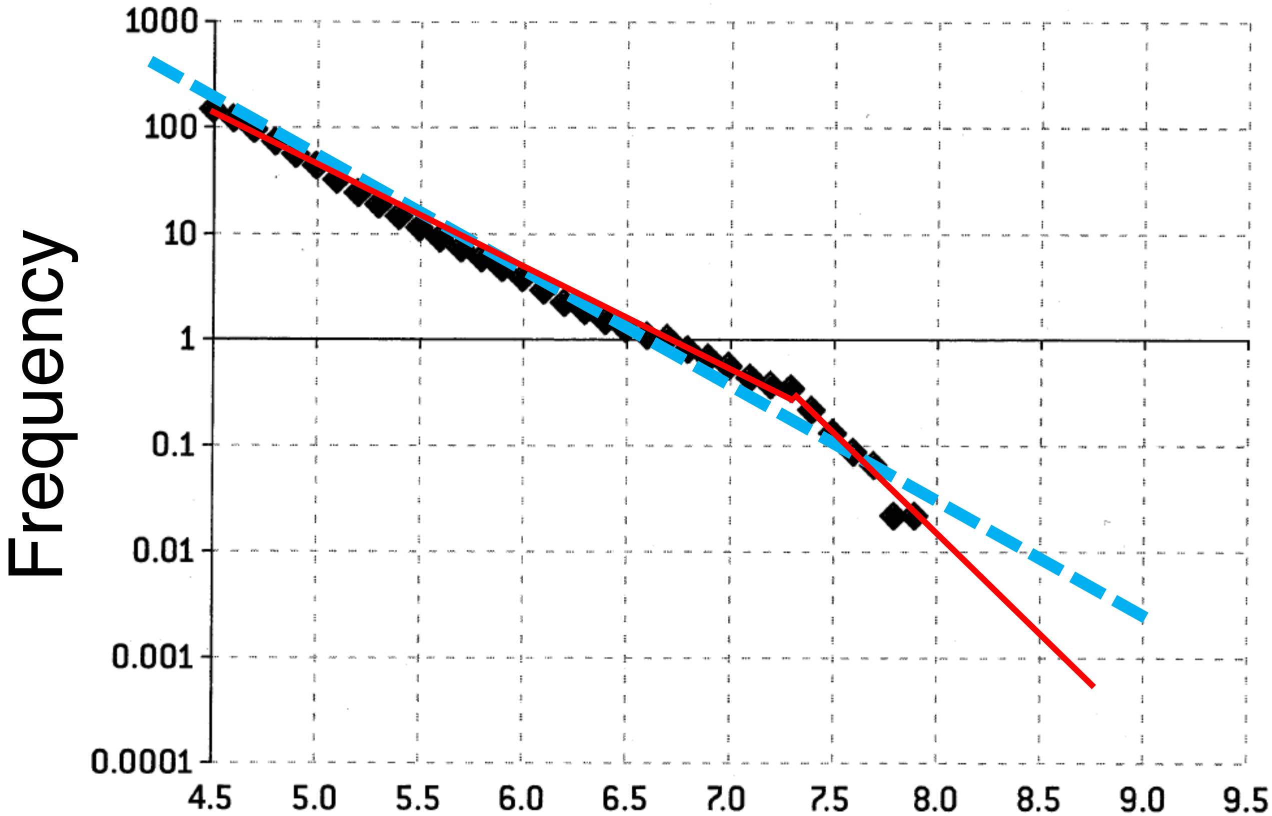
Example: Which is a Better Fit?

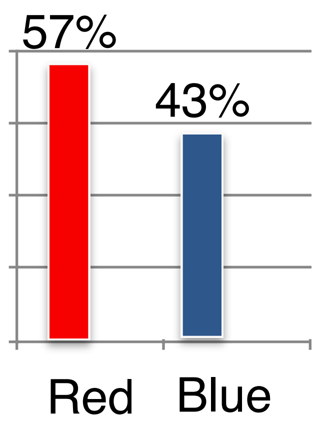


Example: Which is a Better Fit?

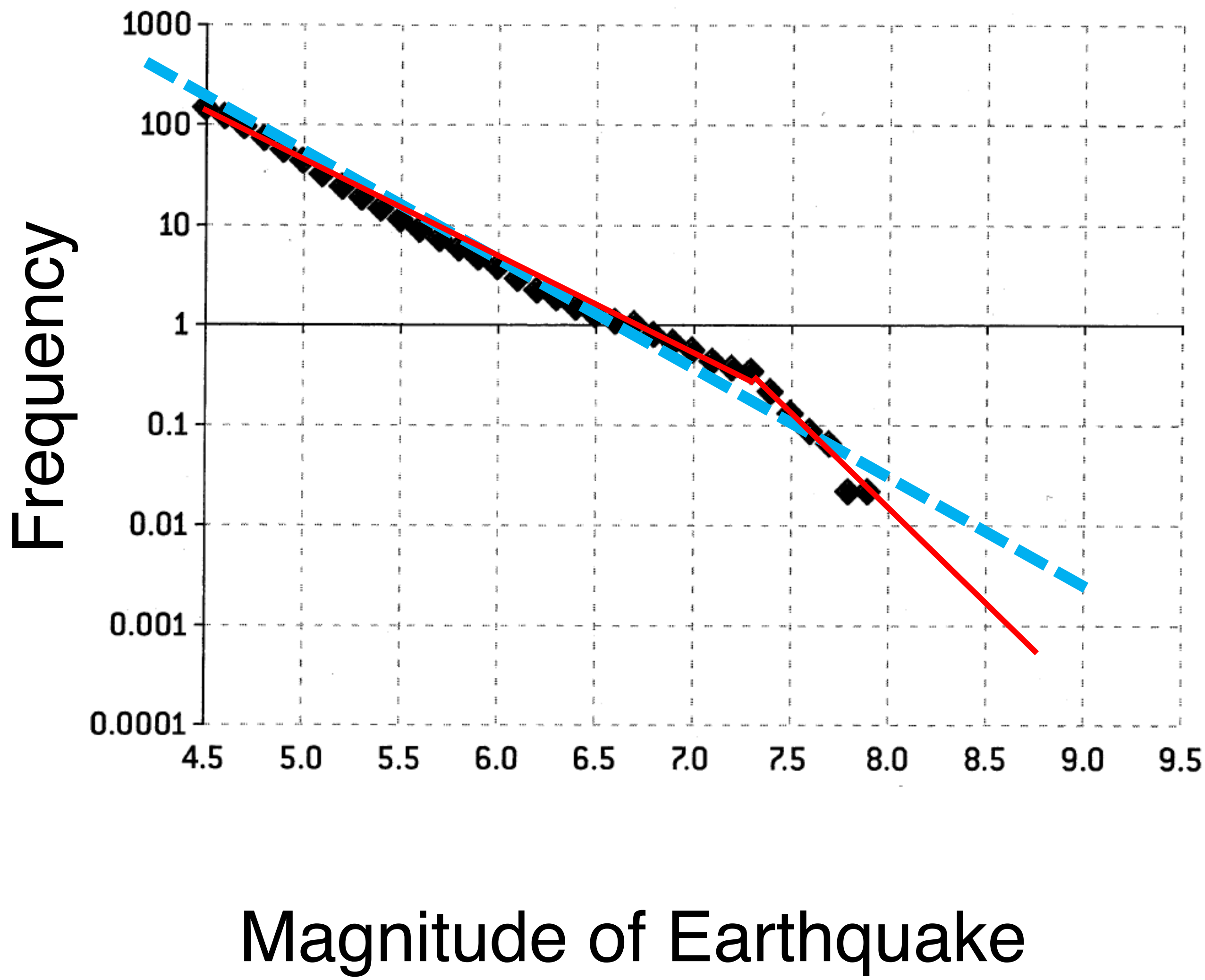


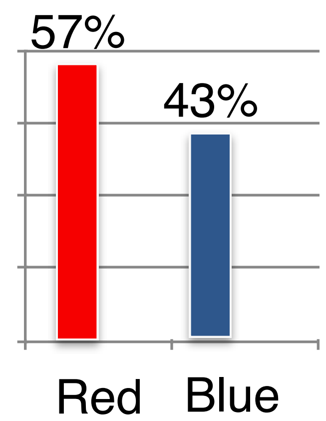
Example: Which is a Better Fit?





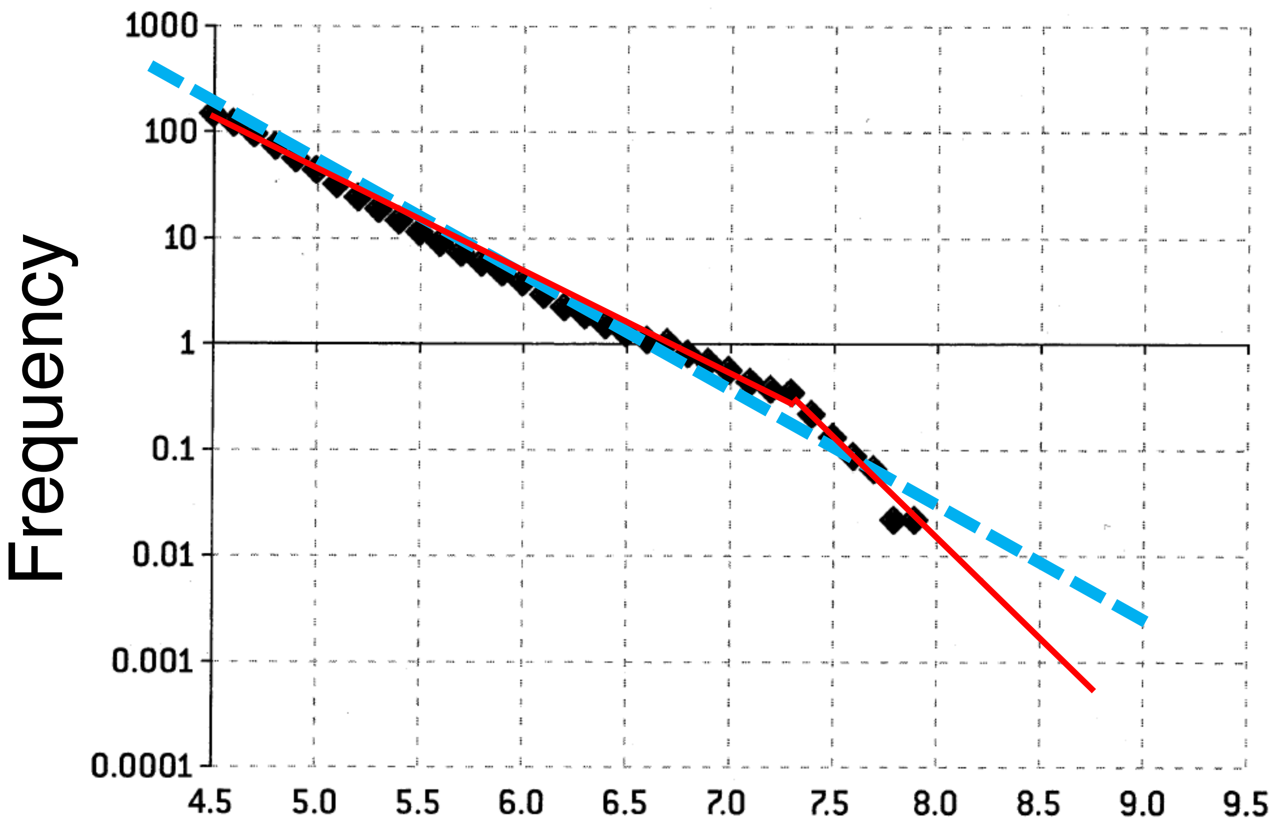
Example: Which is a Better Fit?

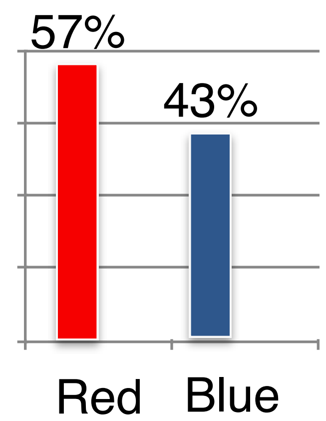




Real data from Tohoku (Japan). Example taken from Nate Silver's book "The signal and noise"

Example: Which is a Better Fit?



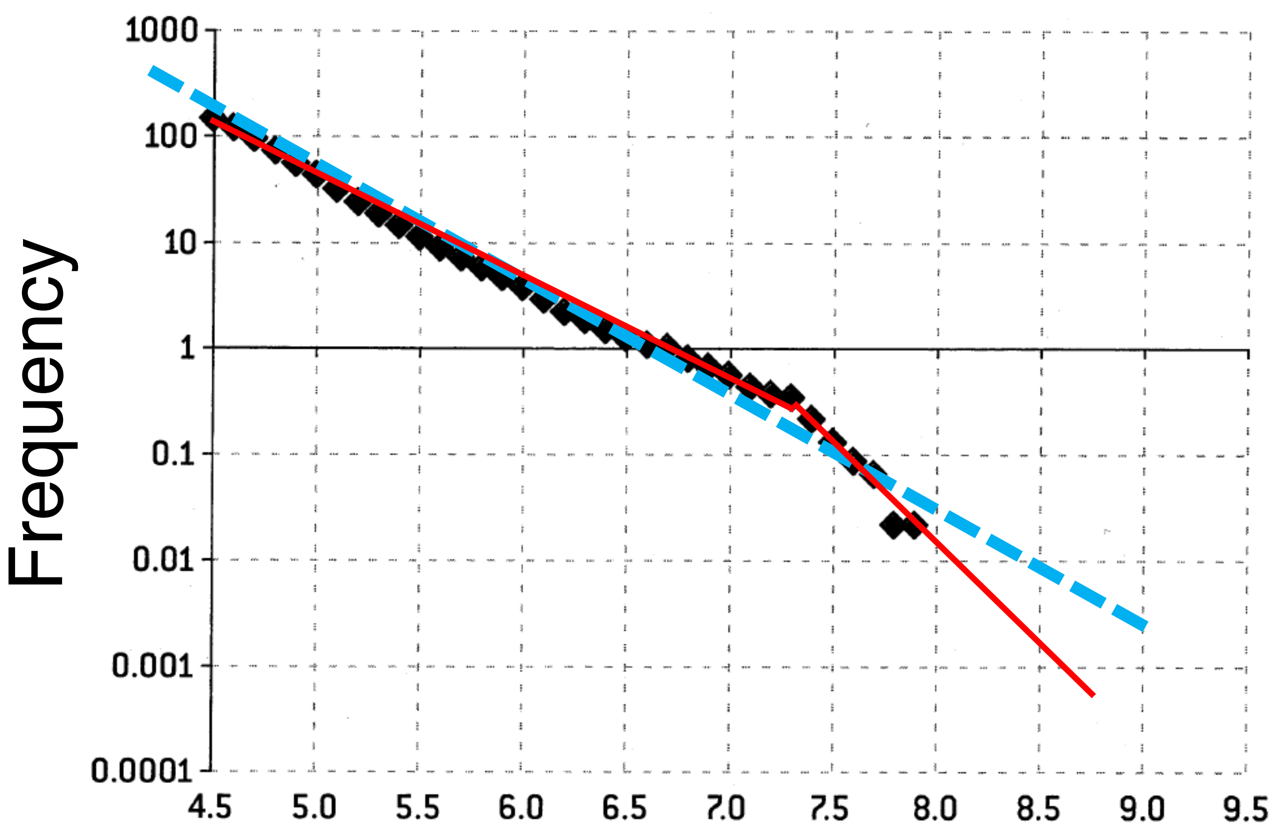


More data → Less data

Magnitude of Earthquake

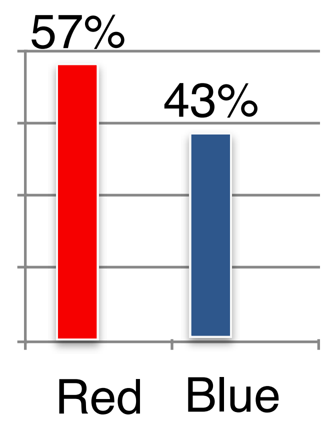
Real data from Tohoku (Japan). Example taken from Nate Silver's book "The signal and noise"

Example: Which is a Better Fit?



More data ⟶ Less data

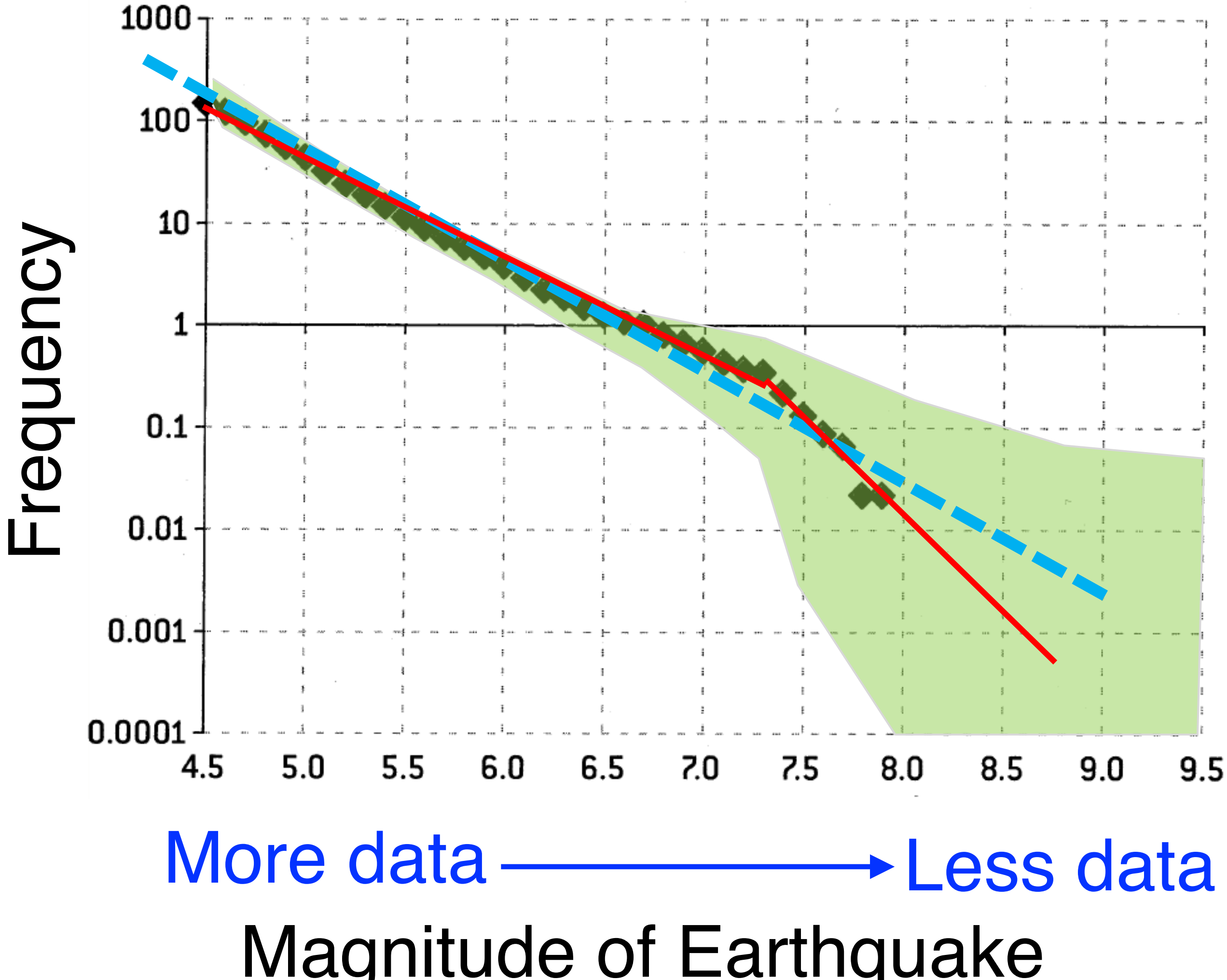
Magnitude of Earthquake



Red is more risky than the blue

Real data from Tohoku (Japan). Example taken from Nate Silver's book "The signal and noise"

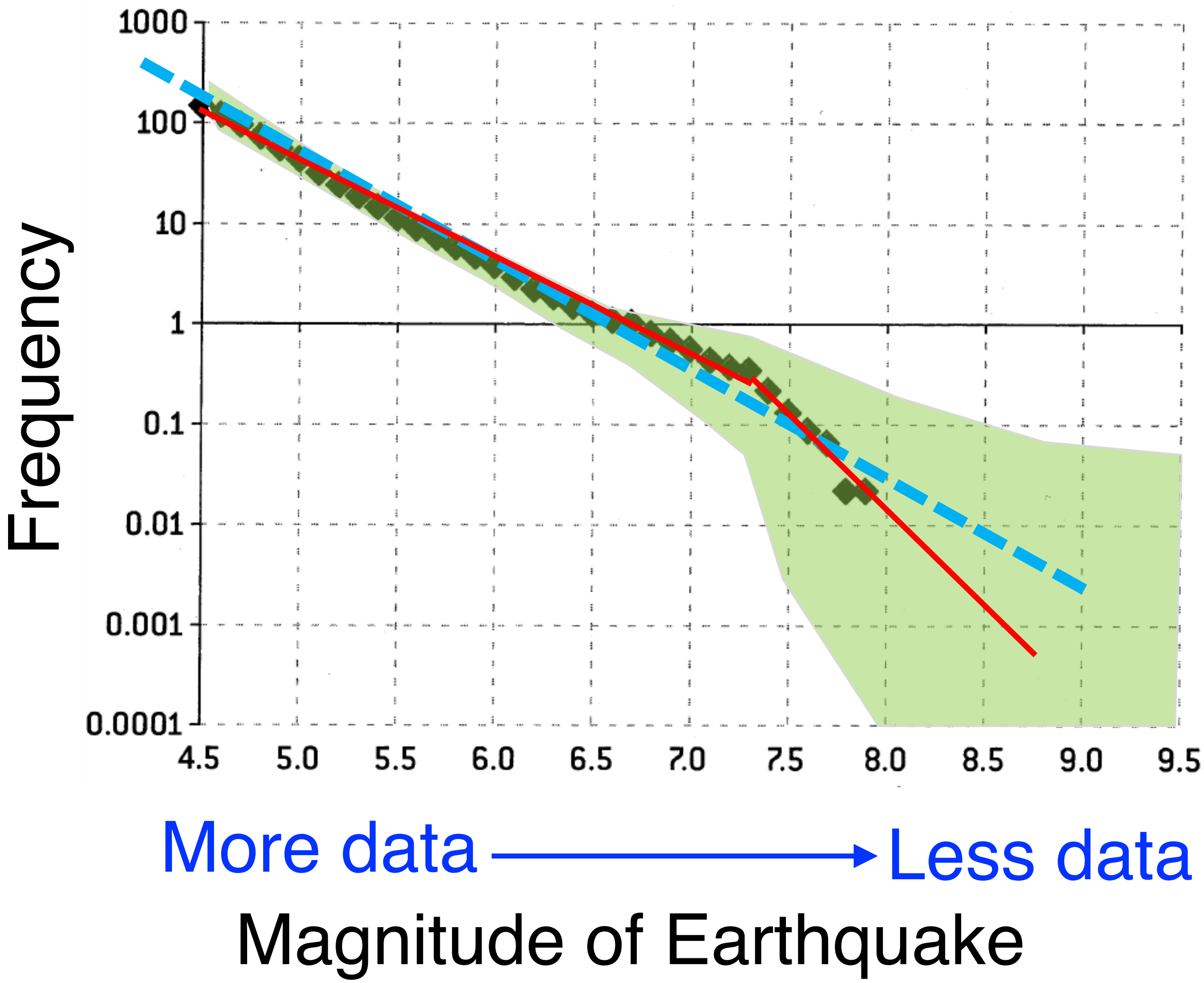
Example: Which is a Better Fit?



Uncertainty: "What the model does not know"

Real data from Tohoku (Japan). Example taken from Nate Silver's book "The signal and noise"

Example: Which is a Better Fit?

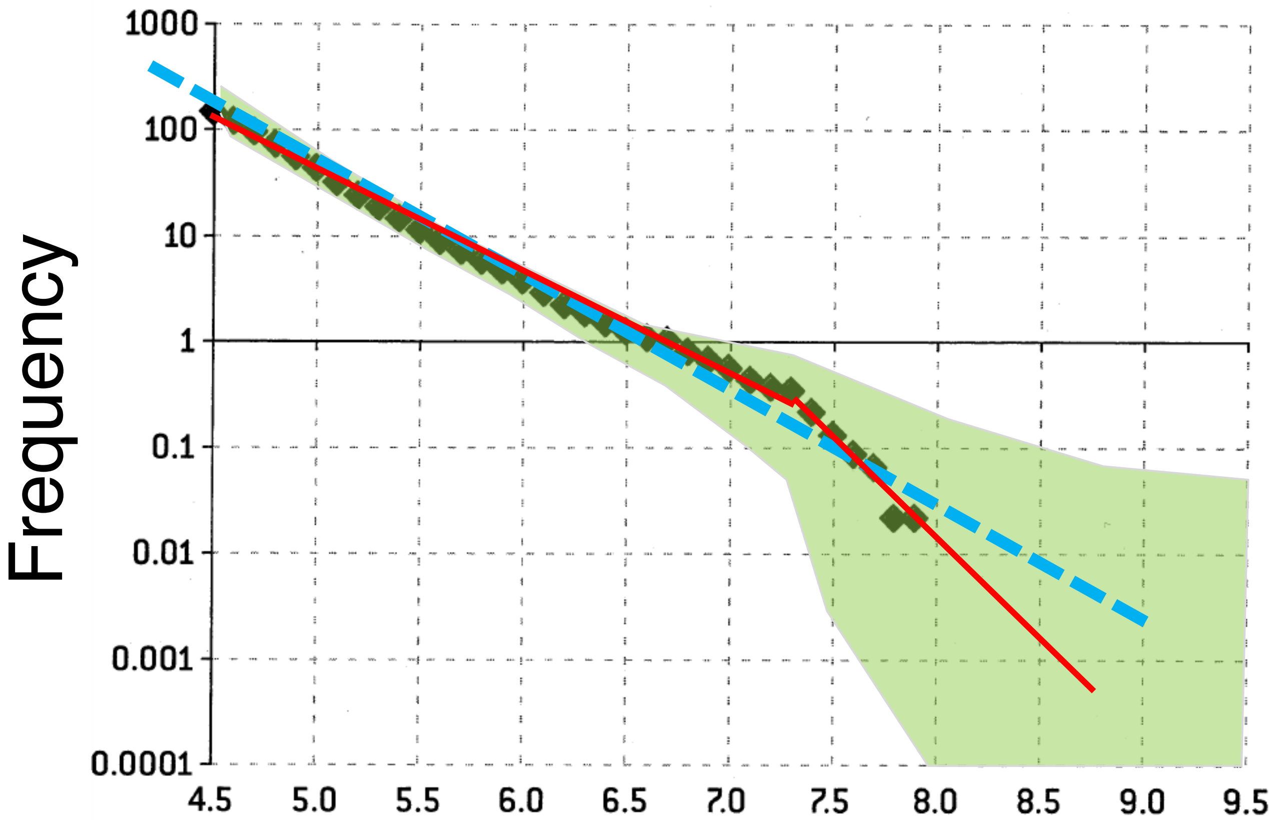


Uncertainty: "What the model does not know"

Choose less risky options!

Real data from Tohoku (Japan). Example taken from Nate Silver's book "The signal and noise"

Example: Which is a Better Fit?



More data → Less data

Magnitude of Earthquake

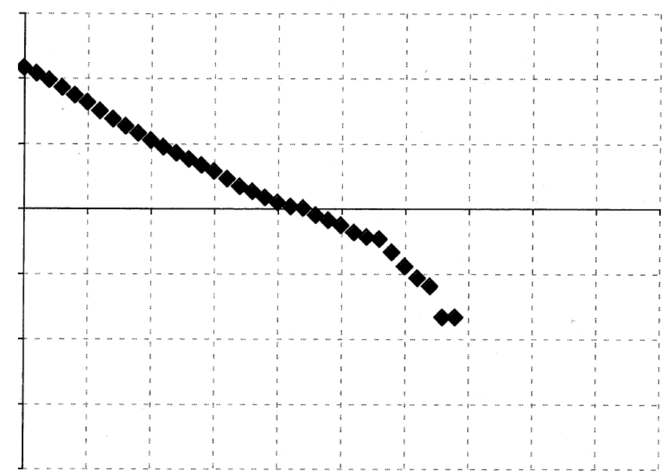
Uncertainty: "What the model does not know"

Choose less risky options!

Avoid data bias with uncertainty!

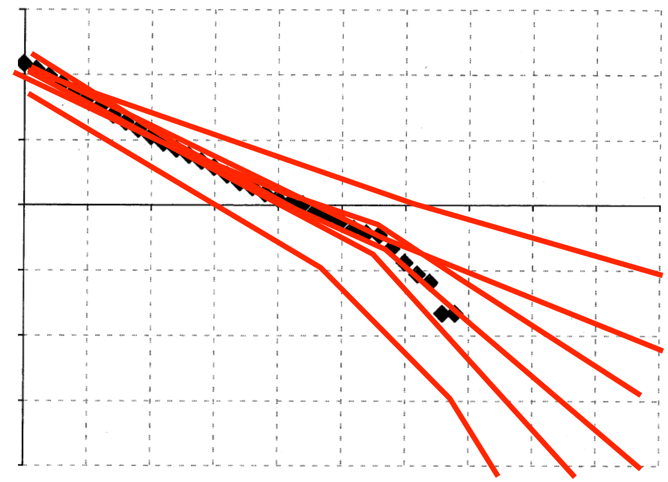
Real data from Tohoku (Japan). Example taken from Nate Silver's book "The signal and noise"

Bayesian Principles



Bayesian Principles

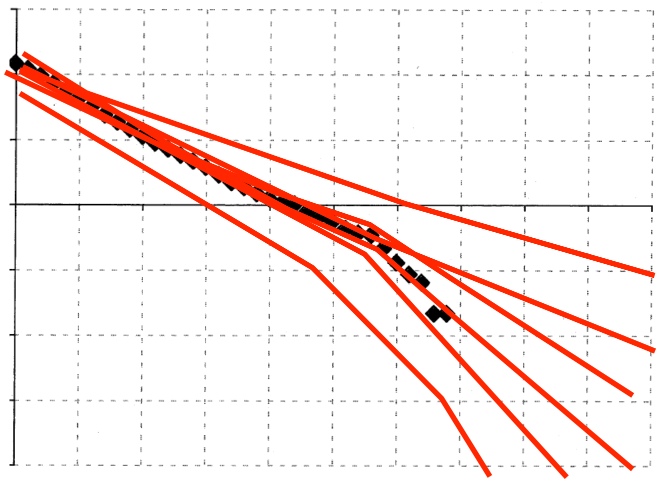
1. Sample $\theta \sim p(\theta)$ prior



Bayesian Principles

1. Sample $\theta \sim p(\theta)$ prior

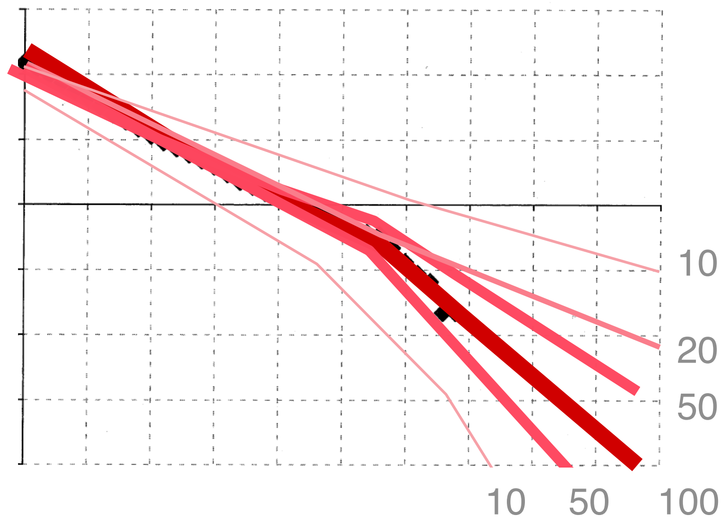
2. Score $p(\mathcal{D}|\theta) = \prod_{i=1}^{N} p(y_i | f_\theta(x_i))$ Likelihood



Bayesian Principles

1. Sample $\theta \sim p(\theta)$ prior

2. Score $p(\mathcal{D}|\theta) = \prod_{i=1}^{N} p(y_i | f_\theta(x_i))$ Likelihood



Bayesian Principles

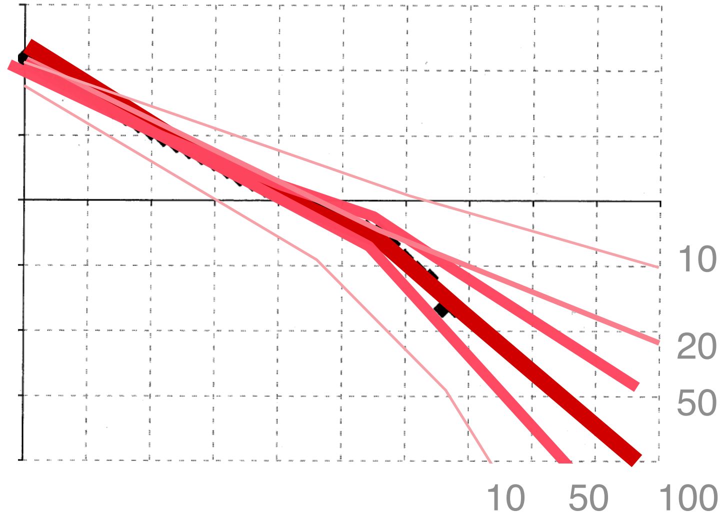
1. Sample $\theta \sim p(\theta)$ prior

2. Score $p(\mathcal{D}|\theta) = \prod_{i=1}^{N} p(y_i|f_\theta(x_i))$ Likelihood

3. Normalize

Posterior Likelihood x Prior

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

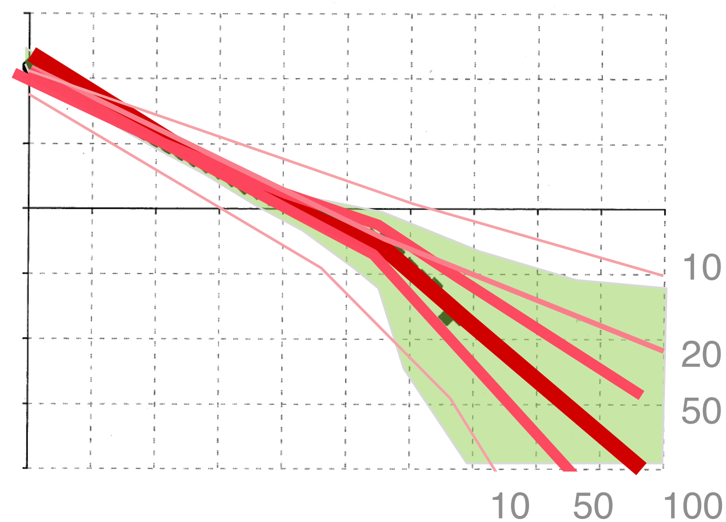


Bayesian Principles

1. Sample $\theta \sim p(\theta)$ prior
2. Score $p(\mathcal{D}|\theta) = \prod_{i=1}^{N} p(y_i | f_\theta(x_i))$ Likelihood
3. Normalize

Posterior Likelihood x Prior

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$



Bayesian Principles

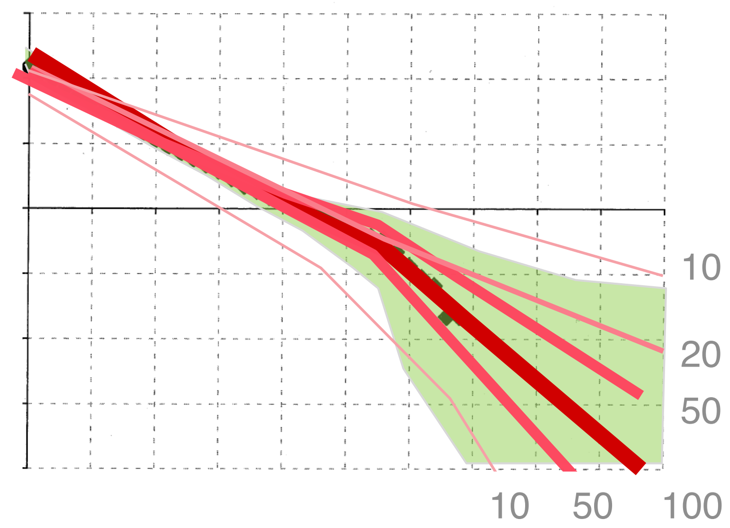
1. Sample $\theta \sim p(\theta)$ prior

2. Score $p(\mathcal{D}|\theta) = \prod_{i=1}^{N} p(y_i | f_\theta(x_i))$ Likelihood

3. Normalize

Posterior Likelihood x Prior

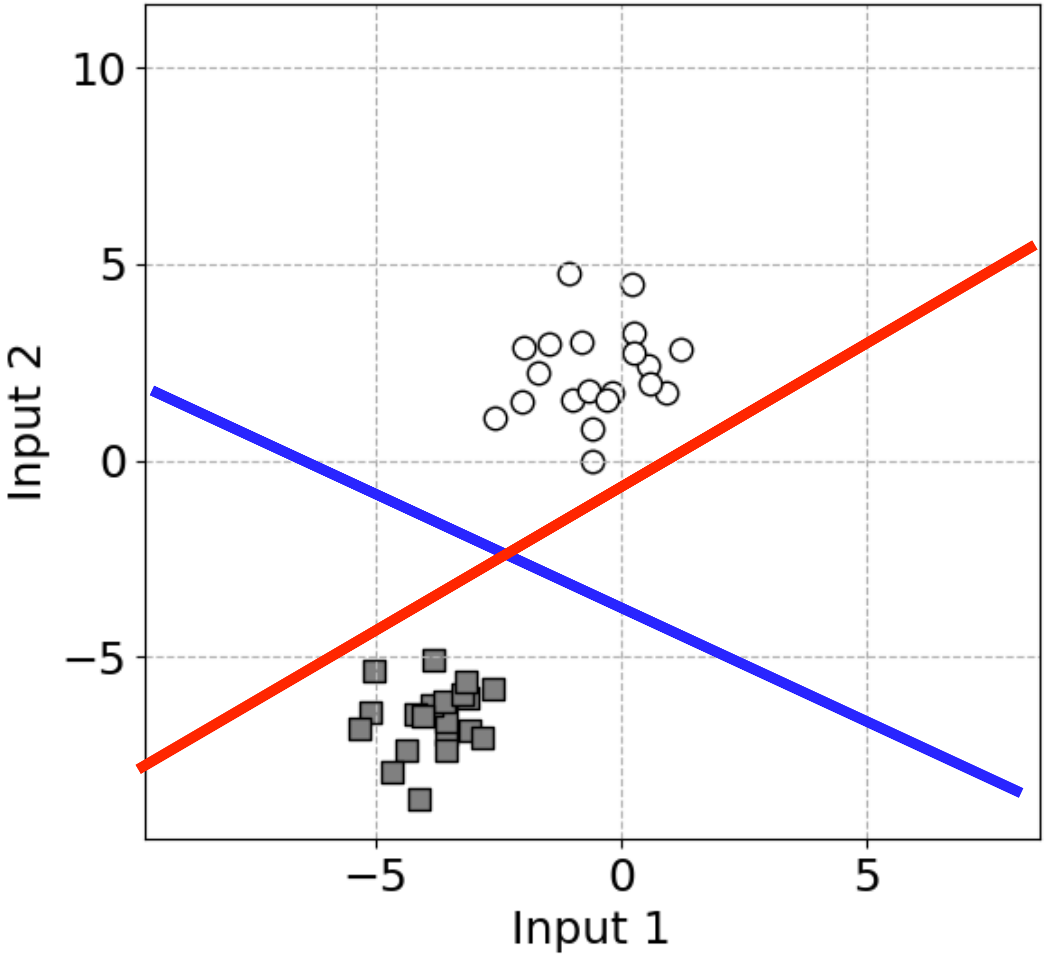
$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$



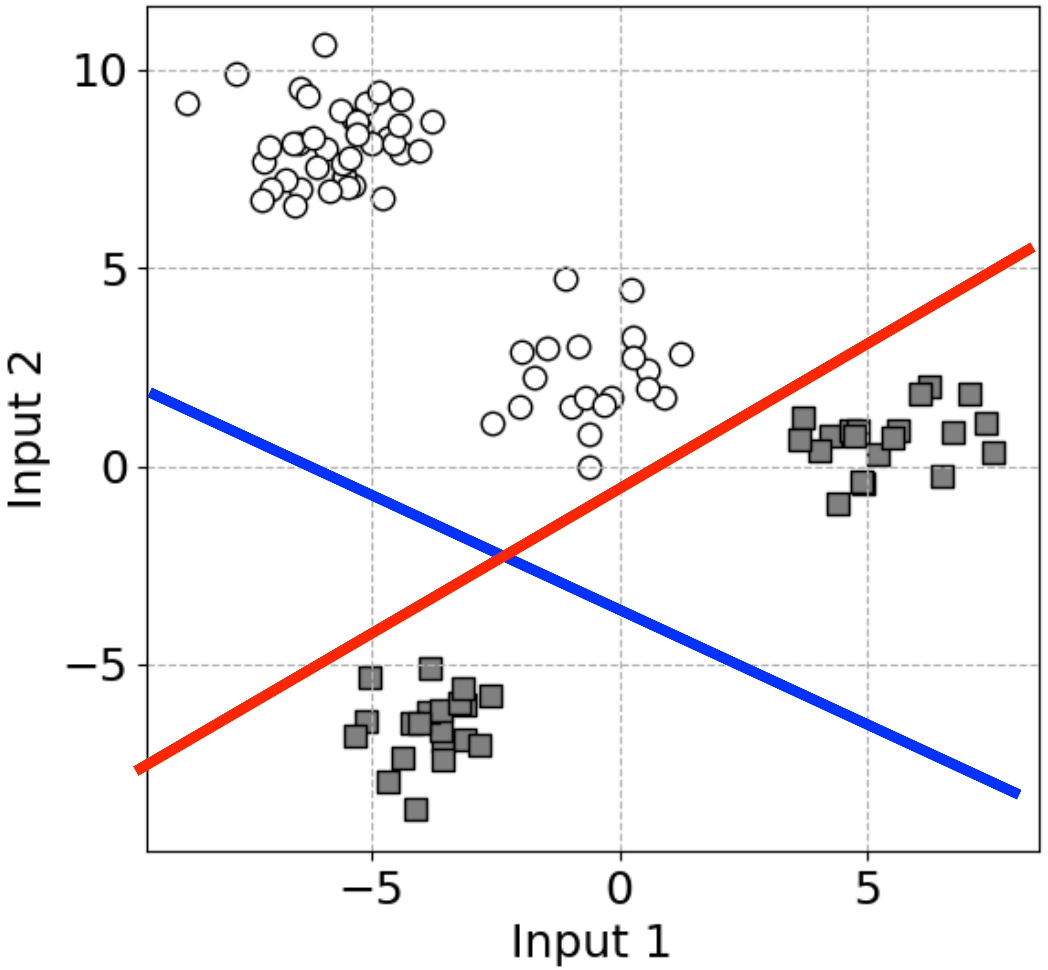
A global method: Integrates over all models

Does not scale to large problem

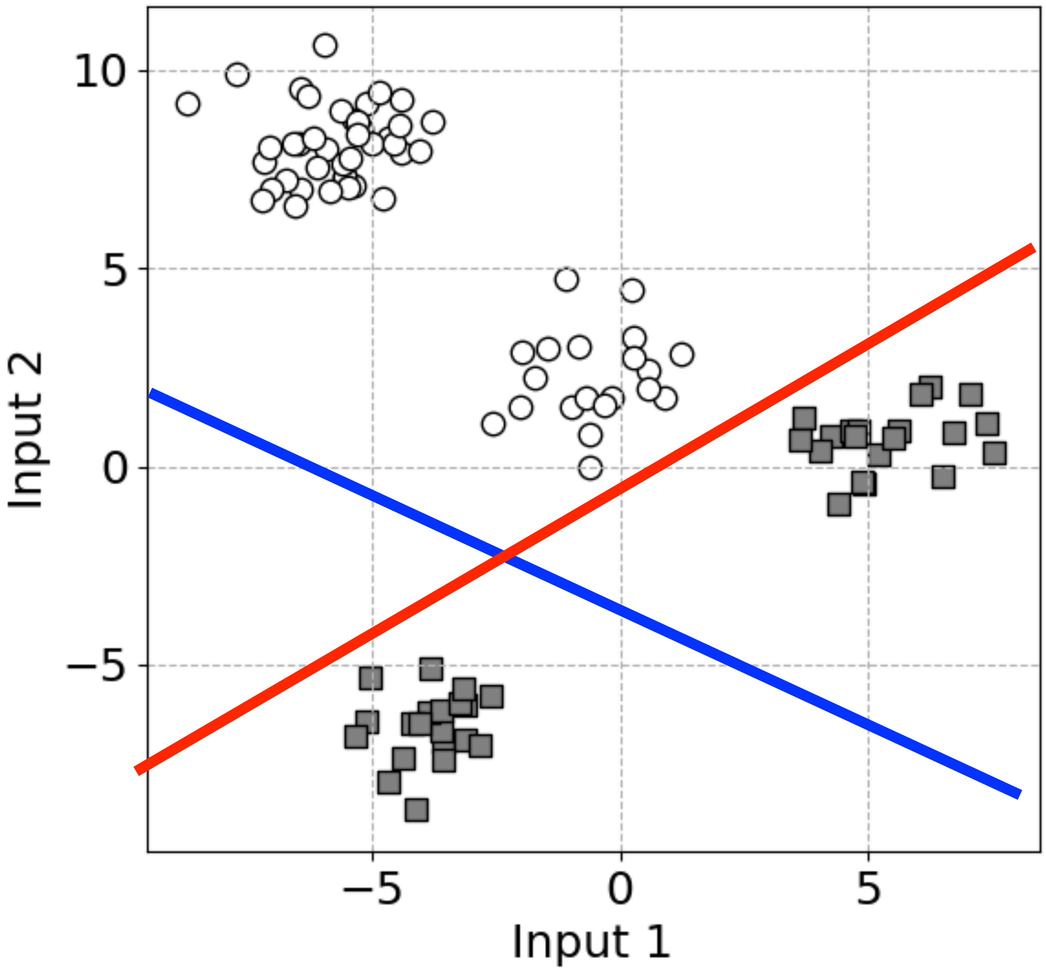
Which is a good classifier?



Which is a good classifier?

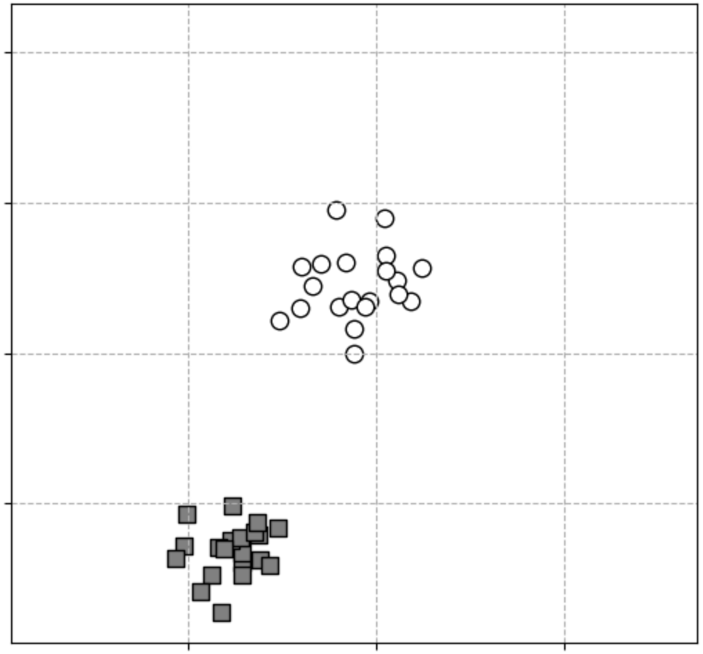


Which is a good classifier?



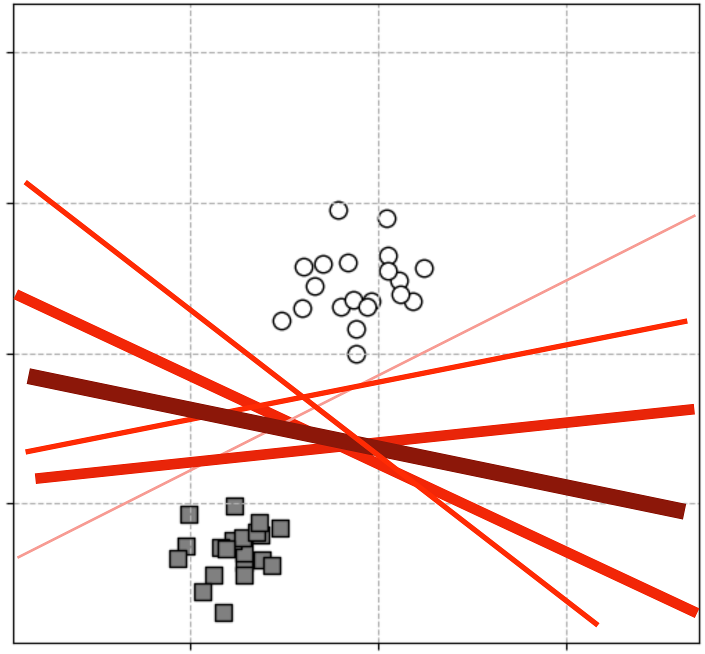
"What the model does not know"

Sequential Bayesian Inference



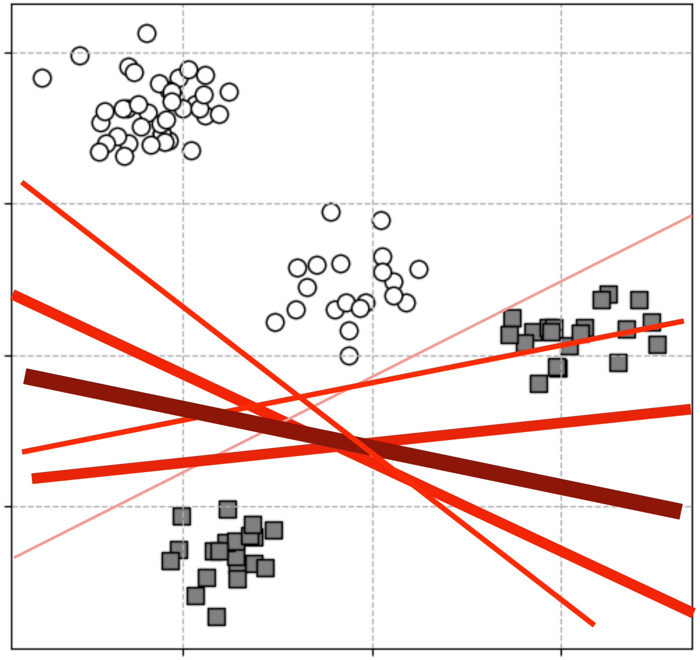
$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)\color{red}{p(\theta)}}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Sequential Bayesian Inference



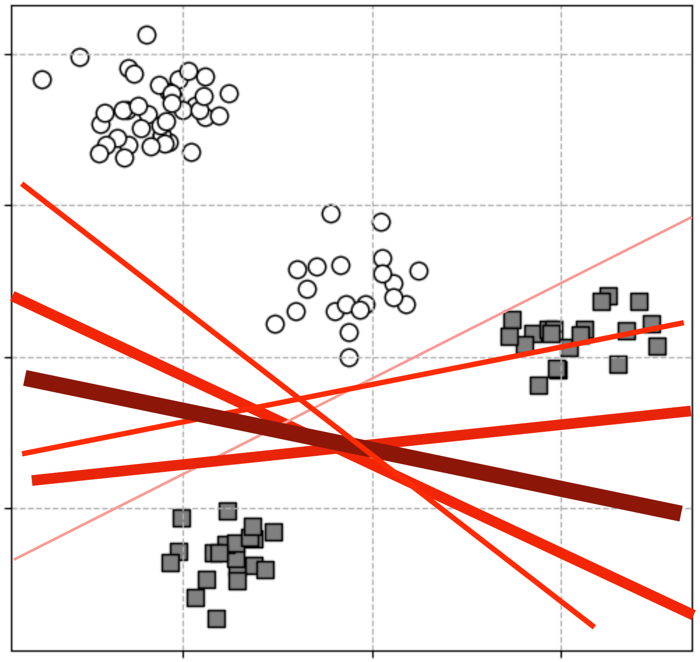
$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)\color{red}{p(\theta)}}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Sequential Bayesian Inference



$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta){\color{red}p(\theta)}}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Sequential Bayesian Inference

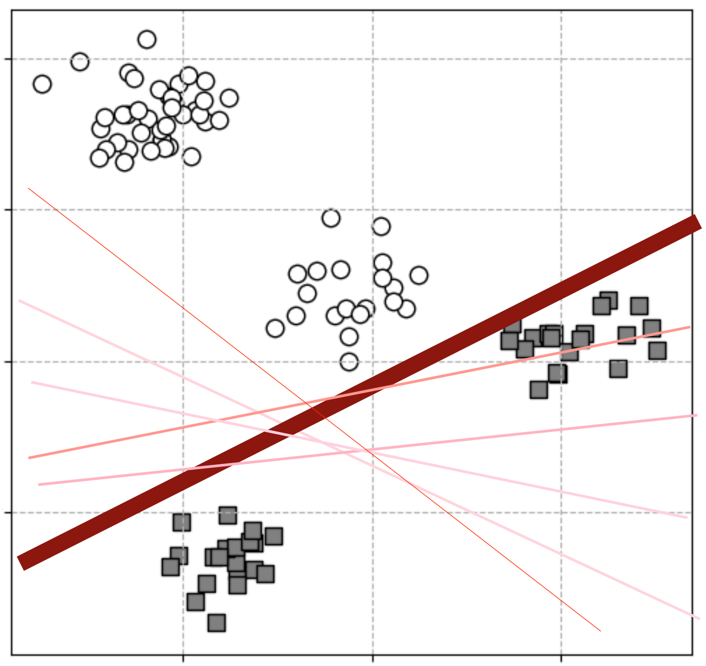


$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)\color{red}{p(\theta)}}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Set the prior to the previous posterior and recompute:

$$p(\theta|\mathcal{D}_2, \mathcal{D}_1) = \frac{p(\mathcal{D}_2|\theta)\color{red}{p(\theta|\mathcal{D}_1)}}{\int p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)d\theta}$$

Sequential Bayesian Inference

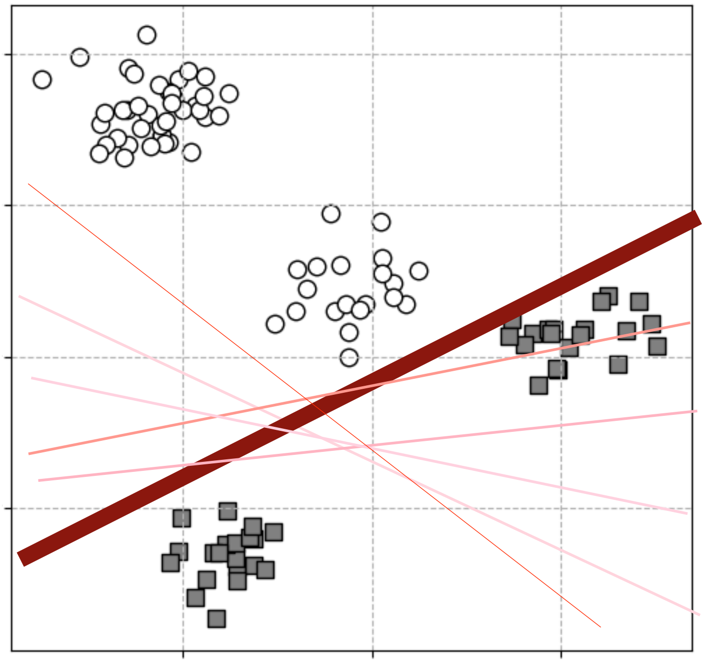


$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Set the prior to the previous posterior and recompute:

$$p(\theta|\mathcal{D}_2, \mathcal{D}_1) = \frac{p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)}{\int p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)d\theta}$$

Sequential Bayesian Inference



$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Set the prior to the previous posterior and recompute:

$$p(\theta|\mathcal{D}_2, \mathcal{D}_1) = \frac{p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)}{\int p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)d\theta}$$

The global property enables sequential update

Bayesian learning

Integration (global)

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

Deep learning

Differentiation (local)

$$\theta \leftarrow \theta - \rho H_\theta^{-1} \nabla_\theta \ell(\theta)$$

	Bayes	DL
Can handle large data and complex models?	✗	✓
Scalable training?	✗	✓
Can estimate uncertainty?	✓	✗
Can perform sequential / active /online / incremental learning?	✓	✗

Deep Learning with Bayesian Principles

- Bayesian principles as common principles
  - By computing “posterior approximations”
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and life-long learning
- Impact: Many learning-algorithms with a common set of principles.

Bayesian Principles

Various types of integrals (approximations)

Bayesian Principles

Various types of integrals (approximations)

Posterior computation: $$q(\theta|\mathcal{D}) \approx \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

Bayesian Principles

Various types of integrals (approximations)

Posterior computation: $$q(\theta|\mathcal{D}) \approx \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

Posterior averaging: $$E(f(\theta)|\mathcal{D}) \approx \int f(\theta)q(\theta|\mathcal{D})d\theta$$

Bayesian Principles

Various types of integrals (approximations)

Posterior computation: $$q(\theta|\mathcal{D}) \approx \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

Posterior averaging: $$E(f(\theta)|\mathcal{D}) \approx \int f(\theta)q(\theta|\mathcal{D})d\theta$$

Posterior marginalization $$q(\theta_i|\mathcal{D}) \approx \int q(\theta|\mathcal{D})d\theta_{/i}$$

Bayesian Principles

Various types of integrals (approximations)

Posterior computation: $$q(\theta|\mathcal{D}) \approx \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

Posterior averaging: $$E(f(\theta)|\mathcal{D}) \approx \int f(\theta)q(\theta|\mathcal{D})d\theta$$

Posterior marginalization $$q(\theta_i|\mathcal{D}) \approx \int q(\theta|\mathcal{D})d\boldsymbol{\theta}_{/i}$$

Our focus is on the first one where we approximate the posterior by solving an optimization problem

Bayesian principles to derive Learning-Algorithms

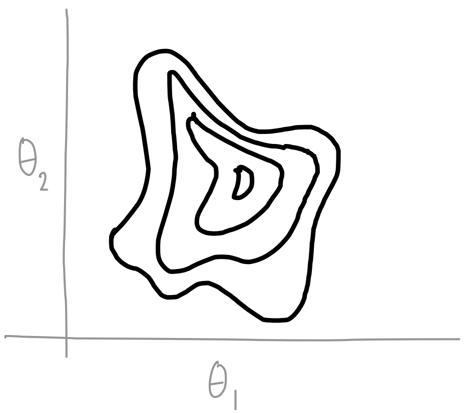
Main ideas: Introduce "posterior approximations" and the "Bayesian learning rule" to estimate them

Complex $\longleftrightarrow$ Simple

Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce "posterior approximations" and the "Bayesian learning rule" to estimate them

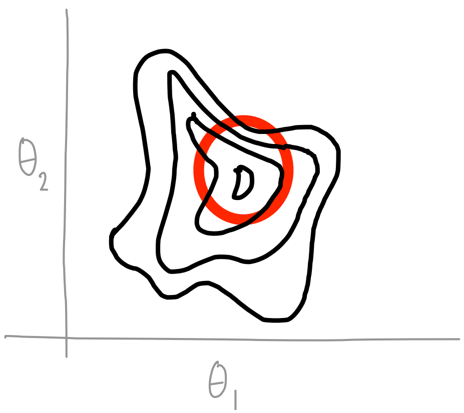
Complex ⟷ Simple



Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce "posterior approximations" and the "Bayesian learning rule" to estimate them

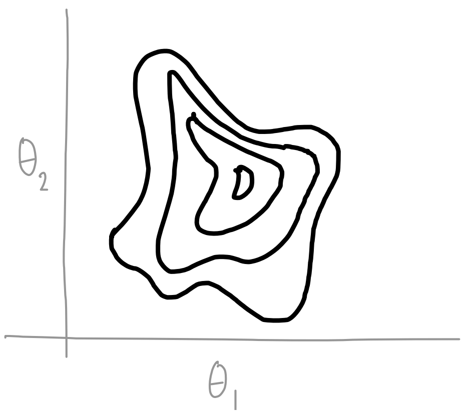
Complex ⟷ Simple

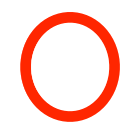


Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce "posterior approximations" and the "Bayesian learning rule" to estimate them

Complex ⟷ Simple

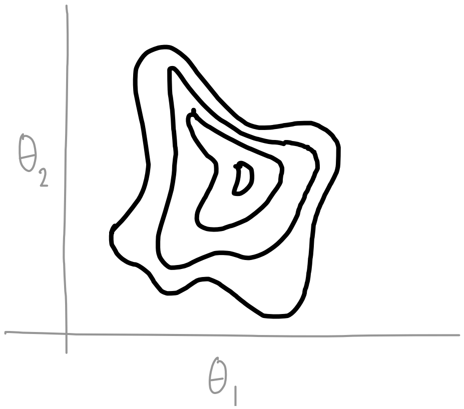


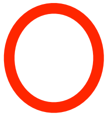


Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them

Complex ⟷ Simple



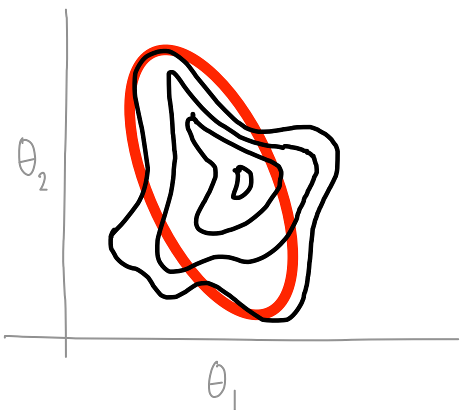


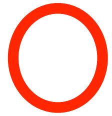
Gradient Descent

Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce "posterior approximations" and the "Bayesian learning rule" to estimate them

Complex ⟷ Simple



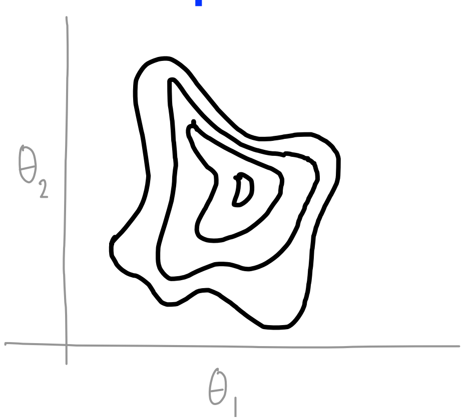


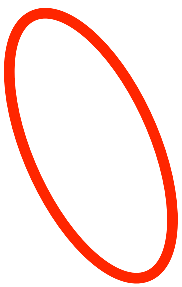
Gradient Descent

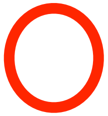
Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce "posterior approximations" and the "Bayesian learning rule" to estimate them

Complex ⟷ Simple



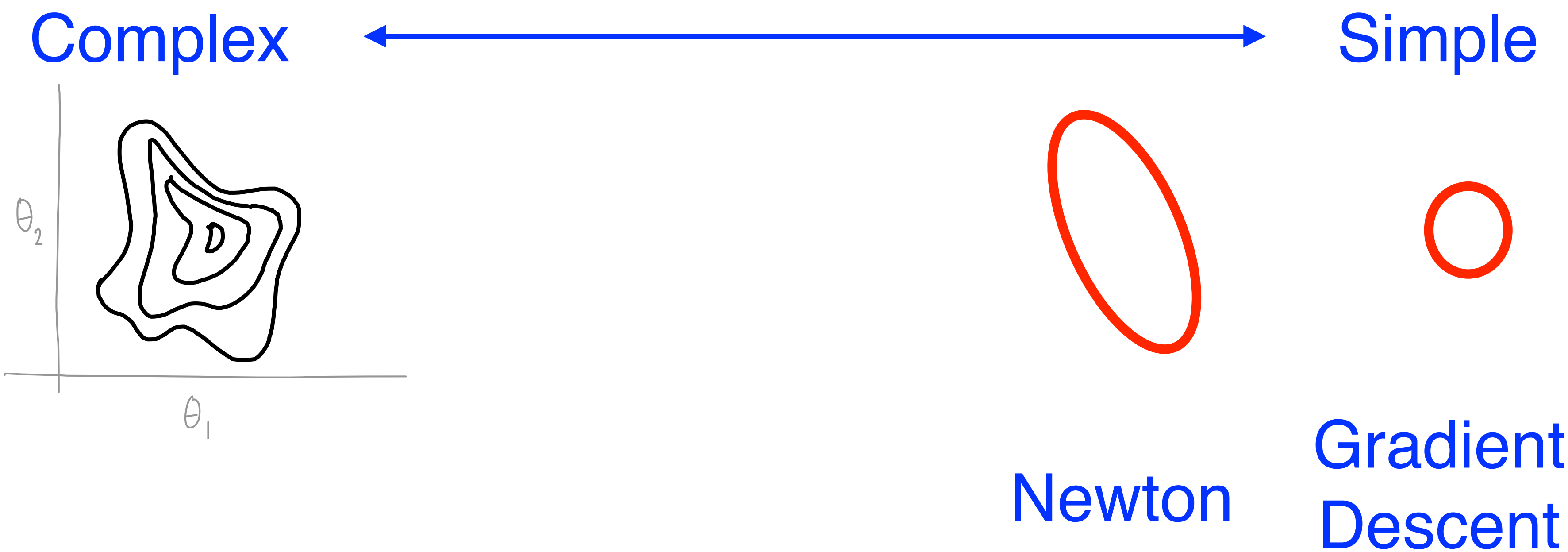




Gradient Descent

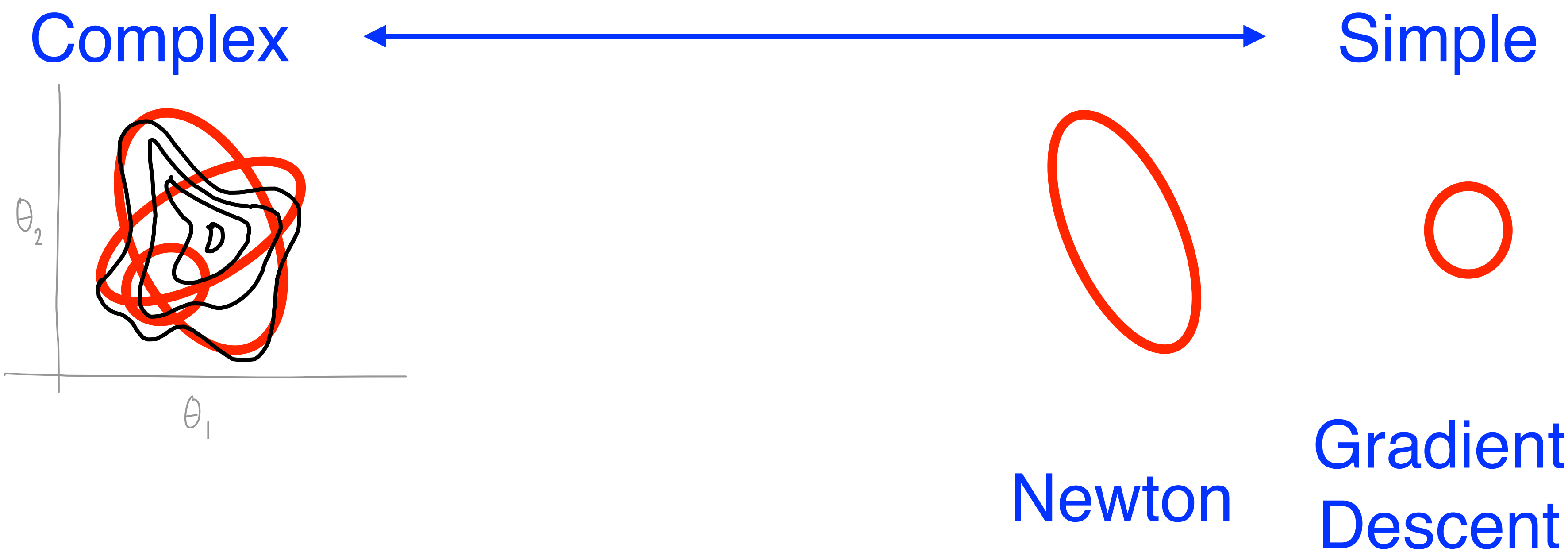
Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce "posterior approximations" and the "Bayesian learning rule" to estimate them



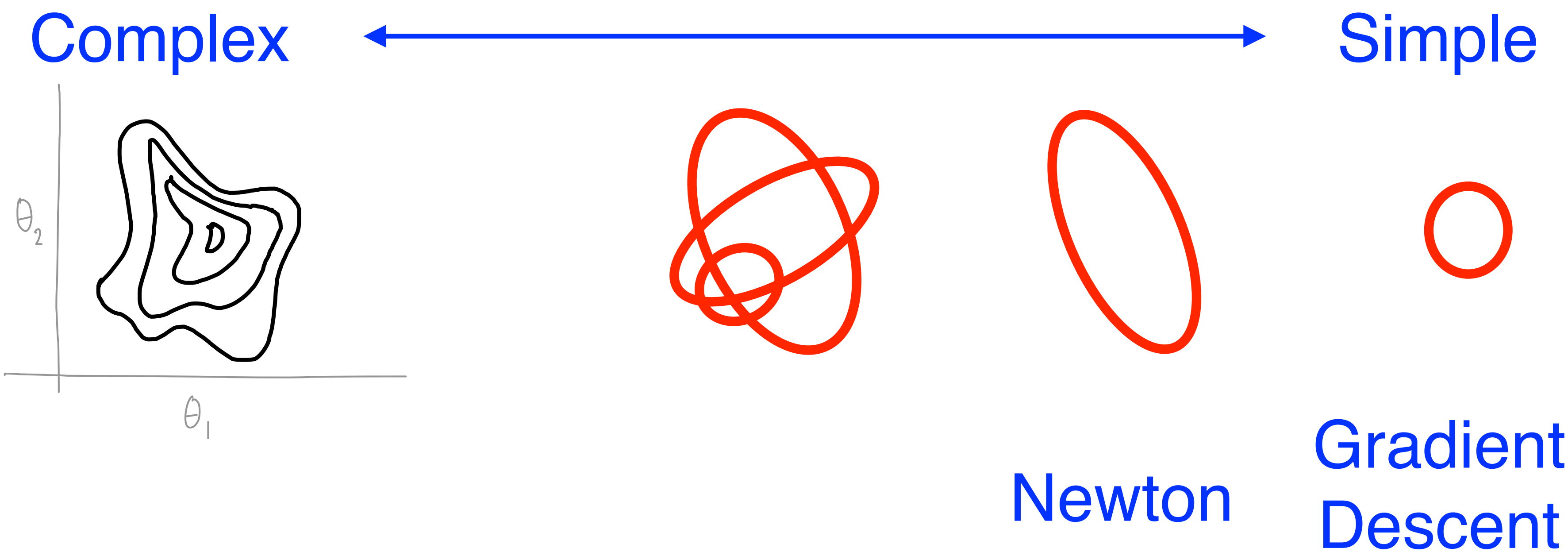
Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them



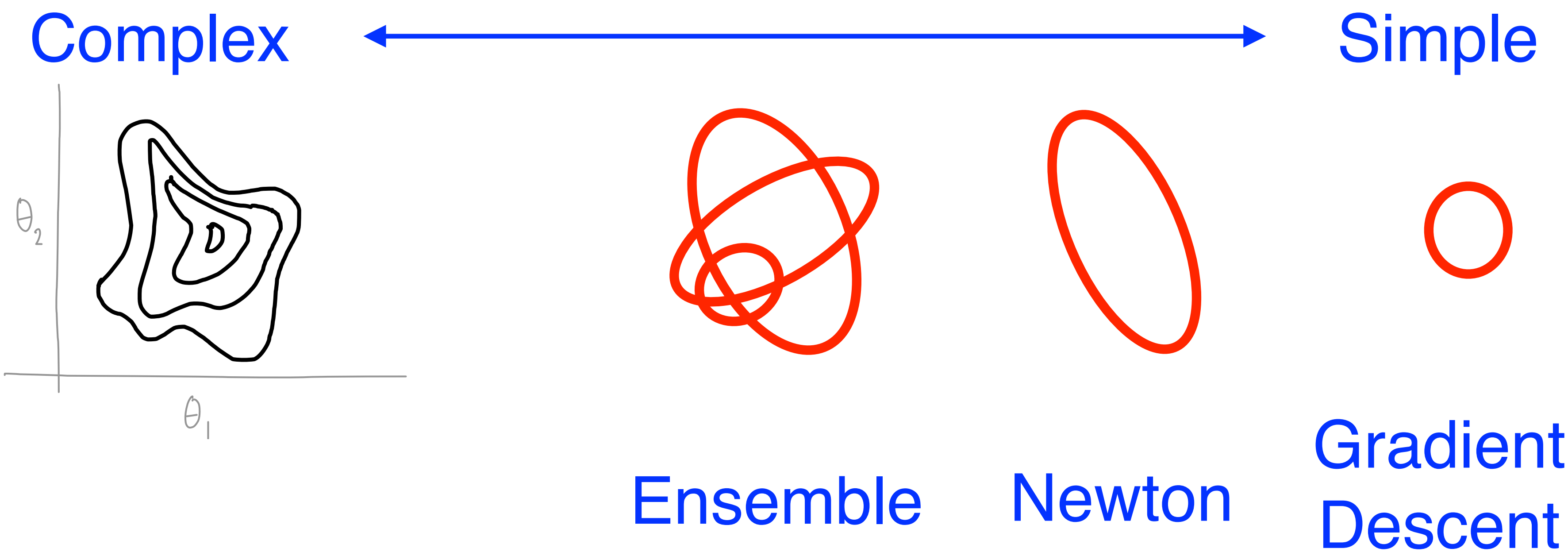
Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce "posterior approximations" and the "Bayesian learning rule" to estimate them



Bayesian principles to derive Learning-Algorithms

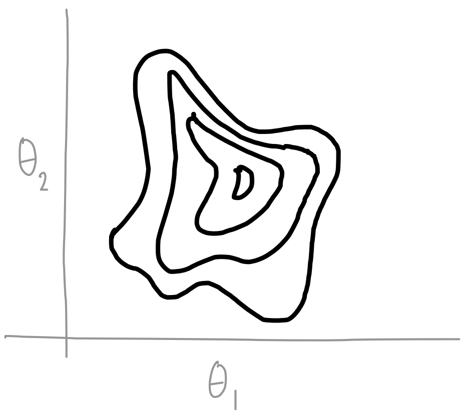
Main ideas: Introduce "posterior approximations" and the "Bayesian learning rule" to estimate them

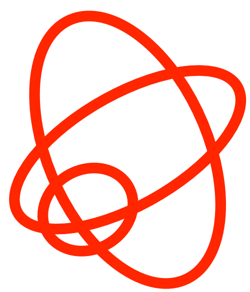


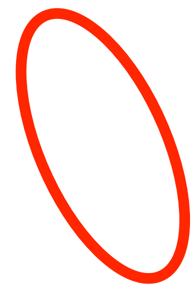
Bayesian principles to derive Learning-Algorithms

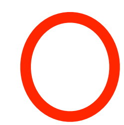
Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them

Complex Simple









Bayes’ rule

Ensemble

Newton

Gradient Descent

Learning-Algorithms from Bayesian Principles

Mohammad Emtiyaz Khan
RIKEN center for Advanced Intelligence Project
Tokyo, Japan

Haavard Rue
CEMSE Division
King Abdullah University of Science and Technology
Thuwal, Saudi Arabia

August 16, 2020
Version 0.7

Abstract

Machine-learning algorithms are commonly derived using ideas from optimization and statistics, followed by an extensive empirical efforts to make them practical as there is a lack of underlying principles to guide this process. In this paper, we present a learning rule derived from Bayesian principles, which enables us to connect a wide-variety of learning algorithms. Using this rule, we can derive a wide-range of learning-algorithms in fields such as probabilistic graphical models, continuous optimization, and deep learning. This includes classical algorithms such as least-squares, Newton's method, and Kalman filter, as well as modern deep-learning algorithms such as stochastic-gradient descent, RMSprop and Adam. Overall, we show that Bayesian principles not only unify, generalize, and improve existing learning-algorithms, but also help us design new ones. [**This is a working draft and a work in progress**]

1. Khan and Rue. "Learning-Algorithms from Bayesian Principles" (2020) (work in progress, an early draft available at https://emtiyaz.github.io/papers/learning_from_bayes.pdf)

Exponential Family Approximations

Natural parameters

Sufficient Statistics

$$q(\theta) \propto \exp\left[\lambda^\top T(\theta)\right]$$

Exponential Family Approximations

Natural parameters

Sufficient Statistics

$$q(\theta) \propto \exp\left[\lambda^\top T(\theta)\right]$$

$$\mathcal{N}(\theta|m, S^{-1}) \propto \exp\left[-\frac{1}{2}(\theta - m)^\top S(\theta - m)\right]$$

Exponential Family Approximations

Natural parameters

Sufficient Statistics

$$q(\theta) \propto \exp\left[\lambda^\top T(\theta)\right]$$

$$\begin{aligned}\mathcal{N}(\theta|m, S^{-1}) &\propto \exp\left[-\frac{1}{2}(\theta - m)^\top S(\theta - m)\right] \\ &\propto \exp\left[(Sm)^\top \theta + \mathrm{Tr}\left(-\frac{S}{2}\theta\theta^\top\right)\right]\end{aligned}$$

Exponential Family Approximations

Natural parameters

Sufficient Statistics

Expectation parameters

$$q(\theta) \propto \exp\left[\lambda^\top T(\theta)\right] \qquad \mu := \mathbb{E}_q[T(\theta)]$$

$$\begin{aligned}\mathcal{N}(\theta|m, S^{-1}) &\propto \exp\left[-\frac{1}{2}(\theta - m)^\top S(\theta - m)\right] \\ &\propto \exp\left[(Sm)^\top \theta + \mathrm{Tr}\left(-\frac{S}{2}\theta\theta^\top\right)\right]\end{aligned}$$

Exponential Family Approximations

Natural parameters — Sufficient Statistics — Expectation parameters

$$q(\theta) \propto \exp\left[\lambda^\top T(\theta)\right] \qquad \mu := \mathbb{E}_q[T(\theta)]$$

$$\mathcal{N}(\theta|m, S^{-1}) \propto \exp\left[-\frac{1}{2}(\theta - m)^\top S(\theta - m)\right]$$
$$\propto \exp\left[(Sm)^\top\theta + \mathrm{Tr}\left(-\frac{S}{2}\theta\theta^\top\right)\right]$$

Gaussian distribution	$q(\theta) := \mathcal{N}(\theta\|m, S^{-1})$
Natural parameters	$\lambda := \{Sm, -S/2\}$
Expectation parameters	$\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^\top)\}$

Bayesian Learning Rule

$$\min_{\theta} \ \ell(\theta) \qquad \text{vs} \qquad \min_{q \in \mathcal{Q}} \ \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

Entropy

1. Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Bayesian Learning Rule

$$\min_{\theta} \ \ell(\theta) \quad \text{vs} \quad \min_{q \in \mathcal{Q}} \ \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

Entropy

Deep Learning algo: $\theta \leftarrow \theta - \rho H_{\theta}^{-1} \nabla_{\theta} \ell(\theta)$

1. Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Bayesian Learning Rule

$$\min_{\theta} \; \ell(\theta) \qquad \text{vs} \qquad \min_{q \in \mathcal{Q}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

Entropy

Deep Learning algo: $\theta \leftarrow \theta - \rho H_\theta^{-1} \nabla_\theta \ell(\theta)$

Bayes learning rule: $\lambda \leftarrow \lambda - \rho \nabla_\mu \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q) \right)$

Natural and Expectation parameters of an exponential family distribution q

1. Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Bayesian Learning Rule

$$\min_{\theta} \ \ell(\theta) \quad \text{vs} \quad \min_{q \in \mathcal{Q}} \ \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

Entropy

Deep Learning algo: $\theta \leftarrow \theta - \rho H_{\theta}^{-1} \nabla_{\theta} \ell(\theta)$

Bayes learning rule: $\lambda \leftarrow \lambda - \rho \nabla_{\mu} \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q) \right)$

Natural and Expectation parameters of
an exponential family distribution q

Deep Learning algorithms can be obtained by

1. Choosing an appropriate approximation q,
2. Giving away the “global” property of the rule

1. Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models.” AIstats (2017).

Deep Learning with Bayesian Principles

- Bayesian principles as common principles
  - By computing “posterior approximations”
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and life-long learning
- Impact: Many learning-algorithms with a common set of principles.

Gradient Descent from Bayes

Gradient descent: $\theta \leftarrow \theta - \rho \nabla_\theta \ell(\theta)$

1. Khan and Rue. “Learning-Algorithms from Bayesian Principles” (2020) (work in progress, an early draft available at https://emtiyaz.github.io/papers/learning_from_bayes.pdf)

Gradient Descent from Bayes

Gradient descent: $\theta \leftarrow \theta - \rho \nabla_\theta \ell(\theta)$

Derived by choosing Gaussian with fixed covariance

Gaussian distribution	$q(\theta) := \mathcal{N}(m, 1)$
Natural parameters	$\lambda := m$
Expectation parameters	$\mu := \mathbb{E}_q[\theta] = m$
Entropy	$\mathcal{H}(q) := \log(2\pi)/2$

1. Khan and Rue. "Learning-Algorithms from Bayesian Principles" (2020) (work in progress, an early draft available at https://emtiyaz.github.io/papers/learning_from_bayes.pdf)

Gradient Descent from Bayes

Gradient descent: $\theta \leftarrow \theta - \rho \nabla_\theta \ell(\theta)$

$$\lambda \leftarrow \lambda - \rho \nabla_{\mu} \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q) \right)$$

Derived by choosing Gaussian with fixed covariance

Gaussian distribution	$q(\theta) := \mathcal{N}(m, 1)$
Natural parameters	$\lambda := m$
Expectation parameters	$\mu := \mathbb{E}_q[\theta] = m$
Entropy	$\mathcal{H}(q) := \log(2\pi)/2$

1. Khan and Rue. "Learning-Algorithms from Bayesian Principles" (2020) (work in progress, an early draft available at https://emtiyaz.github.io/papers/learning_from_bayes.pdf)

Gradient Descent from Bayes

Gradient descent: $\theta \leftarrow \theta - \rho \nabla_\theta \ell(\theta)$

$$m \leftarrow m - \rho \nabla_{m} \mathbb{E}_q[\ell(\theta)]$$

$$\lambda \leftarrow \lambda - \rho \nabla_{\mu} \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q)\right)$$

Derived by choosing Gaussian with fixed covariance

Gaussian distribution	$q(\theta) := \mathcal{N}(m, 1)$
Natural parameters	$\lambda := m$
Expectation parameters	$\mu := \mathbb{E}_q[\theta] = m$
Entropy	$\mathcal{H}(q) := \log(2\pi)/2$

1. Khan and Rue. "Learning-Algorithms from Bayesian Principles" (2020) (work in progress, an early draft available at https://emtiyaz.github.io/papers/learning_from_bayes.pdf)

Gradient Descent from Bayes

Gradient descent: $\theta \leftarrow \theta - \rho \nabla_\theta \ell(\theta)$

"Global" to "local"
$\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$

$$m \leftarrow m - \rho \nabla_m \mathbb{E}_q[\ell(\theta)]$$

$$\lambda \leftarrow \lambda - \rho \nabla_\mu \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q)\right)$$

Derived by choosing Gaussian with fixed covariance

Gaussian distribution $q(\theta) := \mathcal{N}(m, 1)$

Natural parameters $\lambda := m$

Expectation parameters $\mu := \mathbb{E}_q[\theta] = m$

Entropy $\mathcal{H}(q) := \log(2\pi)/2$

1. Khan and Rue. "Learning-Algorithms from Bayesian Principles" (2020) (work in progress, an early draft available at https://emtiyaz.github.io/papers/learning_from_bayes.pdf)

Gradient Descent from Bayes

Gradient descent: $\theta \leftarrow \theta - \rho \nabla_\theta \ell(\theta)$

Bayes Learn Rule: $m \leftarrow m - \rho \nabla_m \ell(m)$

"Global" to "local"
$\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$

$$m \leftarrow m - \rho \nabla_m \mathbb{E}_q[\ell(\theta)]$$

$$\lambda \leftarrow \lambda - \rho \nabla_\mu \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q)\right)$$

Derived by choosing Gaussian with fixed covariance

Gaussian distribution	$q(\theta) := \mathcal{N}(m, 1)$
Natural parameters	$\lambda := m$
Expectation parameters	$\mu := \mathbb{E}_q[\theta] = m$
Entropy	$\mathcal{H}(q) := \log(2\pi)/2$

1. Khan and Rue. "Learning-Algorithms from Bayesian Principles" (2020) (work in progress, an early draft available at https://emtiyaz.github.io/papers/learning_from_bayes.pdf)

Gradient Descent from Bayes

Gradient descent: $\theta \leftarrow \theta - \rho \nabla_\theta \ell(\theta)$

Bayes Learn Rule: $m \leftarrow m - \rho \nabla_m \ell(m)$

"Global" to "local"
$\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$

$$m \leftarrow m - \rho \nabla_m \mathbb{E}_q[\ell(\theta)]$$

$$\lambda \leftarrow \lambda - \rho \nabla_\mu \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q) \right)$$

Derived by choosing Gaussian with fixed covariance

Gaussian distribution	$q(\theta) := \mathcal{N}(m, 1)$
Natural parameters	$\lambda := m$
Expectation parameters	$\mu := \mathbb{E}_q[\theta] = m$
Entropy	$\mathcal{H}(q) := \log(2\pi)/2$

Using stochastic gradients, we get SGD

1. Khan and Rue. "Learning-Algorithms from Bayesian Principles" (2020) (work in progress, an early draft available at https://emtiyaz.github.io/papers/learning_from_bayes.pdf)

Can we obtain Covariances from SGD?

SGD with constant step size = Bayes approx [1,2,3].
So, can we obtain covariances from SGD iterations?

1. Mandt et al. (2017). Stochastic gradient descent as approximate Bayesian inference. JMLR
2. Chaudhari and Soatto (2018). Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. ITA Workshop
3. Maddox et al. (2019). A simple baseline for Bayesian uncertainty in deep learning. NeurIPS.

Can we obtain Covariances from SGD?

SGD with constant step size = Bayes approx [1,2,3].
So, can we obtain covariances from SGD iterations?

SGD estimates only the mean, so the covariance can be arbitrary, e.g., choosing $q(\theta) := \mathcal{N}(m, \Sigma)$ we get

$$m \leftarrow m - \rho \Sigma \nabla_m \ell(m)$$

1. Mandt et al. (2017). Stochastic gradient descent as approximate Bayesian inference. JMLR
2. Chaudhari and Soatto (2018). Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. ITA Workshop
3. Maddox et al. (2019). A simple baseline for Bayesian uncertainty in deep learning. NeurIPS.

Can we obtain Covariances from SGD?

SGD with constant step size = Bayes approx [1,2,3].
So, can we obtain covariances from SGD iterations?

SGD estimates only the mean, so the covariance can be arbitrary, e.g., choosing $q(\theta) := \mathcal{N}(m, \Sigma)$ we get

$$m \leftarrow m - \rho \Sigma \nabla_m \ell(m)$$

Estimation of covariance requires additional computation (essentially the pre-conditioner).

1. Mandt et al. (2017). Stochastic gradient descent as approximate Bayesian inference. JMLR
2. Chaudhari and Soatto (2018). Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. ITA Workshop
3. Maddox et al. (2019). A simple baseline for Bayesian uncertainty in deep learning. NeurIPS.

Newton's Method from Bayes

Newton's method: $\theta \leftarrow \theta - H_\theta^{-1} \left[\nabla_\theta \ell(\theta)\right]$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).

Newton's Method from Bayes

Newton's method: $\theta \leftarrow \theta - H_\theta^{-1} \left[\nabla_\theta \ell(\theta)\right]$

Derived by choosing a multivariate Gaussian

Gaussian distribution	$q(\theta) := \mathcal{N}(\theta \mid m, S^{-1})$
Natural parameters	$\lambda := \{Sm, -S/2\}$
Expectation parameters	$\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^\top)\}$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).

Newton's Method from Bayes

Newton's method: $\theta \leftarrow \theta - H_\theta^{-1} \left[\nabla_\theta \ell(\theta)\right]$

$$\lambda \leftarrow \lambda - \rho \nabla_{\mu} \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q)\right)$$

Derived by choosing a multivariate Gaussian

Gaussian distribution	$q(\theta) := \mathcal{N}(\theta \mid m, S^{-1})$
Natural parameters	$\lambda := \{Sm, -S/2\}$
Expectation parameters	$\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^\top)\}$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).

Newton's Method from Bayes

Newton's method: $\theta \leftarrow \theta - H_\theta^{-1} \left[\nabla_\theta \ell(\theta)\right]$

$$\lambda \leftarrow \lambda - \rho \nabla_{\mu} \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q) \right) \qquad \boxed{-\nabla_\mu \mathcal{H}(q) = \lambda}$$

Derived by choosing a multivariate Gaussian

Gaussian distribution	$q(\theta) := \mathcal{N}(\theta \vert m, S^{-1})$
Natural parameters	$\lambda := \{Sm, -S/2\}$
Expectation parameters	$\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^\top)\}$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).

Newton's Method from Bayes

Newton's method: $\theta \leftarrow \theta - H_\theta^{-1}\left[\nabla_\theta \ell(\theta)\right]$

$$\lambda \leftarrow \lambda - \rho\left(\nabla_{\mu}\mathbb{E}_q[\ell(\theta)] + \lambda\right) \qquad \boxed{-\nabla_\mu \mathcal{H}(q) = \lambda}$$

Derived by choosing a multivariate Gaussian

Gaussian distribution	$q(\theta) := \mathcal{N}(\theta \mid m, S^{-1})$
Natural parameters	$\lambda := \{Sm, -S/2\}$
Expectation parameters	$\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^\top)\}$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).

Newton's Method from Bayes

Newton's method: $\theta \leftarrow \theta - H_\theta^{-1} \left[\nabla_\theta \ell(\theta)\right]$

$$\lambda \leftarrow (1-\rho)\lambda - \rho \nabla_{\mu} \mathbb{E}_q[\ell(\theta)] \qquad \boxed{-\nabla_\mu \mathcal{H}(q) = \lambda}$$

Derived by choosing a multivariate Gaussian

Gaussian distribution	$q(\theta) := \mathcal{N}(\theta \vert m, S^{-1})$
Natural parameters	$\lambda := \{Sm, -S/2\}$
Expectation parameters	$\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^\top)\}$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).

Newton's Method from Bayes

Newton's method: $\theta \leftarrow \theta - H_\theta^{-1}\left[\nabla_\theta \ell(\theta)\right]$

$$Sm \leftarrow (1-\rho)Sm - \rho\nabla_{\mathbb{E}_q(\theta)}\mathbb{E}_q[\ell(\theta)]$$

$$\lambda \leftarrow (1-\rho)\lambda - \rho\nabla_{\mu}\mathbb{E}_q[\ell(\theta)]$$

$$-\nabla_\mu \mathcal{H}(q) = \lambda$$

Derived by choosing a multivariate Gaussian

Gaussian distribution $q(\theta) := \mathcal{N}(\theta|m, S^{-1})$

Natural parameters $\lambda := \{Sm, -S/2\}$

Expectation parameters $\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^\top)\}$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).

Newton's Method from Bayes

Newton's method: $\theta \leftarrow \theta - H_\theta^{-1}\left[\nabla_\theta \ell(\theta)\right]$

$$Sm \leftarrow (1-\rho)Sm - \rho \nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)]$$

$$-\frac{1}{2}S \leftarrow -(1-\rho)\frac{1}{2}S + \rho \nabla_{\mathbb{E}_q(\theta\theta^\top)} \mathbb{E}_q[\ell(\theta)]$$

$$\lambda \leftarrow (1-\rho)\lambda - \rho \nabla_\mu \mathbb{E}_q[\ell(\theta)]$$

$$\boxed{-\nabla_\mu \mathcal{H}(q) = \lambda}$$

Derived by choosing a multivariate Gaussian

Gaussian distribution $q(\theta) := \mathcal{N}(\theta | m, S^{-1})$

Natural parameters $\lambda := \{Sm, -S/2\}$

Expectation parameters $\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^\top)\}$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).

Newton's Method from Bayes

Newton's method: $\theta \leftarrow \theta - H_\theta^{-1}\left[\nabla_\theta \ell(\theta)\right]$

$$Sm \leftarrow (1-\rho)Sm - \rho \nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)]$$

$$S \leftarrow (1-\rho)S - \rho 2 \nabla_{\mathbb{E}_q(\theta\theta^\top)} \mathbb{E}_q[\ell(\theta)]$$

$$\lambda \leftarrow (1-\rho)\lambda - \rho \nabla_{\mu} \mathbb{E}_q[\ell(\theta)]$$

$$-\nabla_\mu \mathcal{H}(q) = \lambda$$

Derived by choosing a multivariate Gaussian

Gaussian distribution $q(\theta) := \mathcal{N}(\theta | m, S^{-1})$

Natural parameters $\lambda := \{Sm, -S/2\}$

Expectation parameters $\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^\top)\}$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).

Newton's Method from Bayes

Newton's method: $\theta \leftarrow \theta - H_\theta^{-1} \left[\nabla_\theta \ell(\theta)\right]$

$$\boxed{\begin{aligned} Sm &\leftarrow (1-\rho)Sm - \rho \nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)] \\ S &\leftarrow (1-\rho)S - \rho 2 \nabla_{\mathbb{E}_q(\theta\theta^\top)} \mathbb{E}_q[\ell(\theta)] \end{aligned}}$$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).

Newton's Method from Bayes

Newton's method: $\theta \leftarrow \theta - H_\theta^{-1} \left[\nabla_\theta \ell(\theta)\right]$

Express in terms of gradient and Hessian of loss:

$$\nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[\nabla_\theta \ell(\theta)] - 2\mathbb{E}_q[H_\theta]m$$

$$\nabla_{\mathbb{E}_q(\theta\theta^\top)} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[H_\theta]$$

$$\boxed{\begin{aligned} Sm &\leftarrow (1-\rho)Sm - \rho\nabla_{\mathbb{E}_q(\theta)}\mathbb{E}_q[\ell(\theta)] \\ S &\leftarrow (1-\rho)S - \rho 2\nabla_{\mathbb{E}_q(\theta\theta^\top)}\mathbb{E}_q[\ell(\theta)] \end{aligned}}$$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).

Newton's Method from Bayes

Newton's method: $\theta \leftarrow \theta - H_\theta^{-1}\left[\nabla_\theta \ell(\theta)\right]$

"Global" to "local"
$\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$

Express in terms of gradient and Hessian of loss:

$$\nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[\nabla_\theta \ell(\theta)] - 2\mathbb{E}_q[H_\theta] m$$

$$\nabla_{\mathbb{E}_q(\theta\theta^\top)} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[H_\theta]$$

$$Sm \leftarrow (1-\rho) Sm - \rho \nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)]$$
$$S \leftarrow (1-\rho) S - \rho 2 \nabla_{\mathbb{E}_q(\theta\theta^\top)} \mathbb{E}_q[\ell(\theta)]$$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).

Newton's Method from Bayes

Newton's method: $\theta \leftarrow \theta - H_\theta^{-1} \left[\nabla_\theta \ell(\theta)\right]$

$$m \leftarrow m - \rho S^{-1} \nabla_m \ell(m)$$
$$S \leftarrow (1-\rho) S + \rho H_m$$

"Global" to "local"
$$\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$$

Express in terms of gradient and Hessian of loss:

$$\nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[\nabla_\theta \ell(\theta)] - 2\mathbb{E}_q[H_\theta] m$$

$$\nabla_{\mathbb{E}_q(\theta\theta^\top)} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[H_\theta]$$

$$Sm \leftarrow (1-\rho) Sm - \rho \nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)]$$
$$S \leftarrow (1-\rho) S - \rho 2 \nabla_{\mathbb{E}_q(\theta\theta^\top)} \mathbb{E}_q[\ell(\theta)]$$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).

Newton's Method from Bayes

Newton's method: $\theta \leftarrow \theta - H_\theta^{-1}\left[\nabla_\theta \ell(\theta)\right]$

Set ρ =1 to get $\quad m \leftarrow m - H_m^{-1}[\nabla_m \ell(m)]$

$$m \leftarrow m - \rho S^{-1} \nabla_m \ell(m)$$
$$S \leftarrow (1-\rho) S + \rho H_m$$

"Global" to "local"
$\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$

Express in terms of gradient and Hessian of loss:

$$\nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[\nabla_\theta \ell(\theta)] - 2\mathbb{E}_q[H_\theta] m$$

$$\nabla_{\mathbb{E}_q(\theta\theta^\top)} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[H_\theta]$$

$$Sm \leftarrow (1-\rho) Sm - \rho \nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)]$$
$$S \leftarrow (1-\rho) S - \rho 2 \nabla_{\mathbb{E}_q(\theta\theta^\top)} \mathbb{E}_q[\ell(\theta)]$$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).

Gaussians Approximation by Second-order Optimization

To estimate $q(\theta) := \mathcal{N}(\theta|m, S^{-1})$

$$m \leftarrow m - \rho S^{-1} \nabla_m \ell(m)$$

$$S \leftarrow (1 - \rho) S + \rho H_m$$

Estimate of mean requires 1st-order information

Estimate of covariance requires 2nd-order information [1]

1. Opper and Archambeau, C. (2009). The variational Gaussian approximation revisited. Neural Computation, 21(3):786–792.

Gaussians Approximation by Second-order Optimization

To estimate $q(\theta) := \mathcal{N}(\theta|m, S^{-1})$

$$m \leftarrow m - \rho S^{-1} \nabla_m \ell(m)$$

$$S \leftarrow (1-\rho) S + \rho H_m$$

Estimate of mean requires 1st-order information
Estimate of covariance requires 2nd-order information [1]

Can't escape this principle, but can reduce the computation with heuristics and approximations!

1. Opper and Archambeau, C. (2009). The variational Gaussian approximation revisited. Neural Computation, 21(3):786–792.

Optimization vs Bayes

What is the difference between the solutions?

$$m \leftarrow m - \rho S^{-1} \mathbb{E}_q [\nabla_\theta \ell(\theta)]$$

$$S \leftarrow (1-\rho) S + \rho \mathbb{E}_q [H_\theta]$$

Optimization vs Bayes

What is the difference between the solutions?

$$m \leftarrow m - \rho S^{-1} \mathbb{E}_q[\nabla_\theta \ell(\theta)]$$
$$S \leftarrow (1-\rho) S + \rho \mathbb{E}_q[H_\theta]$$

$$\Rightarrow$$

$$\mathbb{E}_{q_*}[\nabla_\theta \ell(\theta)] = 0$$
$$S_* = \mathbb{E}_{q_*}[H_\theta]$$

Optimization vs Bayes

What is the difference between the solutions?

$$m \leftarrow m - \rho S^{-1} \mathbb{E}_q[\nabla_\theta \ell(\theta)] \quad \Rightarrow \quad \mathbb{E}_{q_*}[\nabla_\theta \ell(\theta)] = 0$$

$$S \leftarrow (1-\rho) S + \rho \mathbb{E}_q[H_\theta] \quad \Rightarrow \quad S_* = \mathbb{E}_{q_*}[H_\theta]$$

The optimality conditions are different for q* and theta*

	Bayes	Optimization
1st-order:		
2nd-order:		

Optimization vs Bayes

What is the difference between the solutions?

$$m \leftarrow m - \rho S^{-1} \mathbb{E}_q[\nabla_\theta \ell(\theta)]$$
$$S \leftarrow (1-\rho) S + \rho \mathbb{E}_q[H_\theta]$$

$$\Rightarrow$$

$$\mathbb{E}_{q_*}[\nabla_\theta \ell(\theta)] = 0$$
$$S_* = \mathbb{E}_{q_*}[H_\theta]$$

The optimality conditions are different for q* and theta*

	Bayes	Optimization
1st-order:	$\mathbb{E}_{q_*}[\nabla_\theta \ell(\theta)] = 0$	
2nd-order:	$\mathbb{E}_{q_*}[H_\theta] \succ 0$	

Optimization vs Bayes

What is the difference between the solutions?

$$m \leftarrow m - \rho S^{-1}\mathbb{E}_q[\nabla_\theta \ell(\theta)] \qquad \mathbb{E}_{q_*}[\nabla_\theta \ell(\theta)] = 0$$

$$S \leftarrow (1-\rho)S + \rho\mathbb{E}_q[H_\theta] \qquad S_* = \mathbb{E}_{q_*}[H_\theta]$$

The optimality conditions are different for q* and theta*

	Bayes	Optimization
1st-order:	$\mathbb{E}_{q_*}[\nabla_\theta \ell(\theta)] = 0$	$\nabla_\theta \ell(\theta_*)] = 0$
2nd-order:	$\mathbb{E}_{q_*}[H_\theta] \succ 0$	$H_{\theta_*} \succ 0$

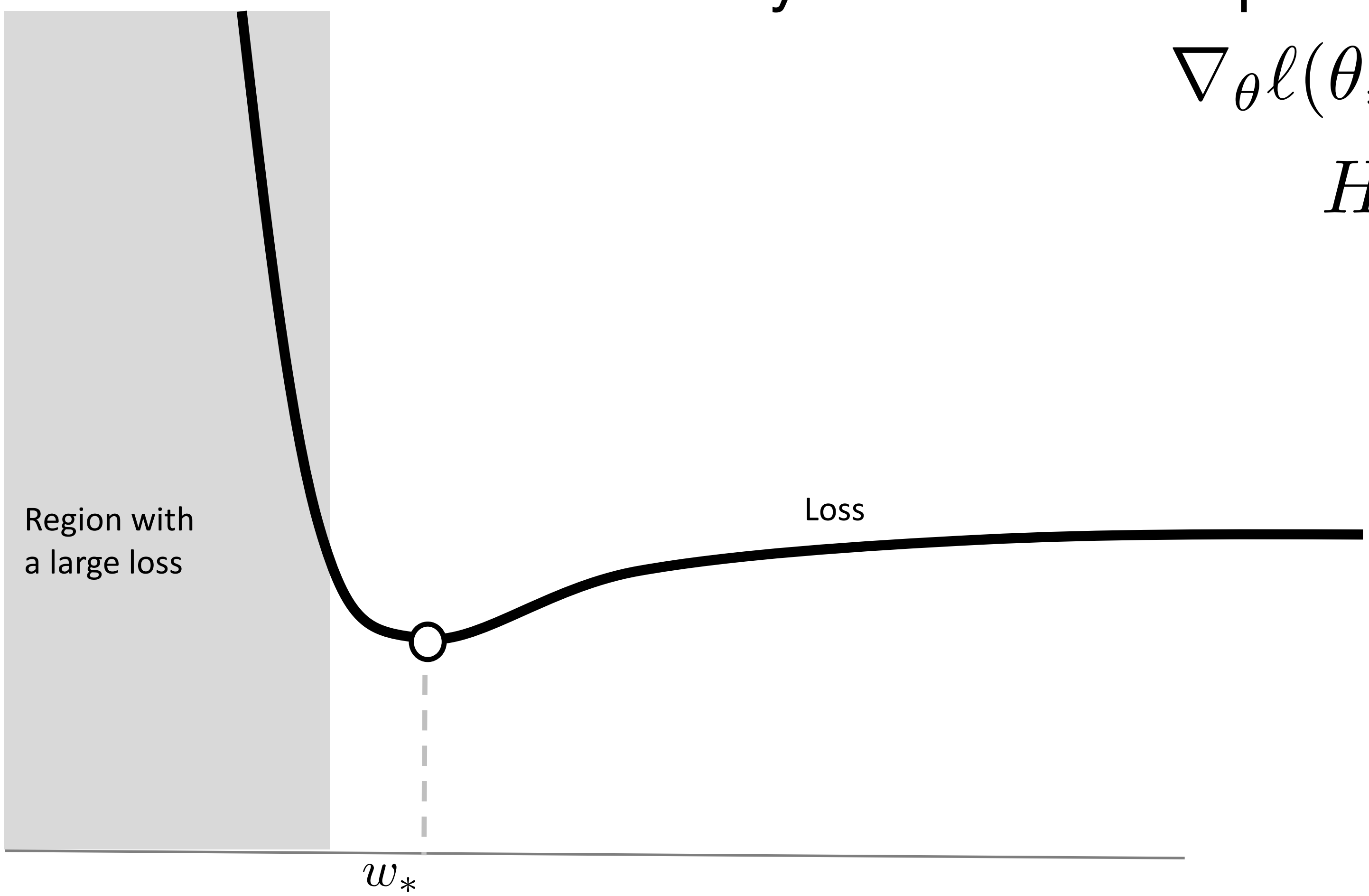
The Optimization Solution

Bayes

Optimization

$$\nabla_\theta \ell(\theta_*)] = 0$$

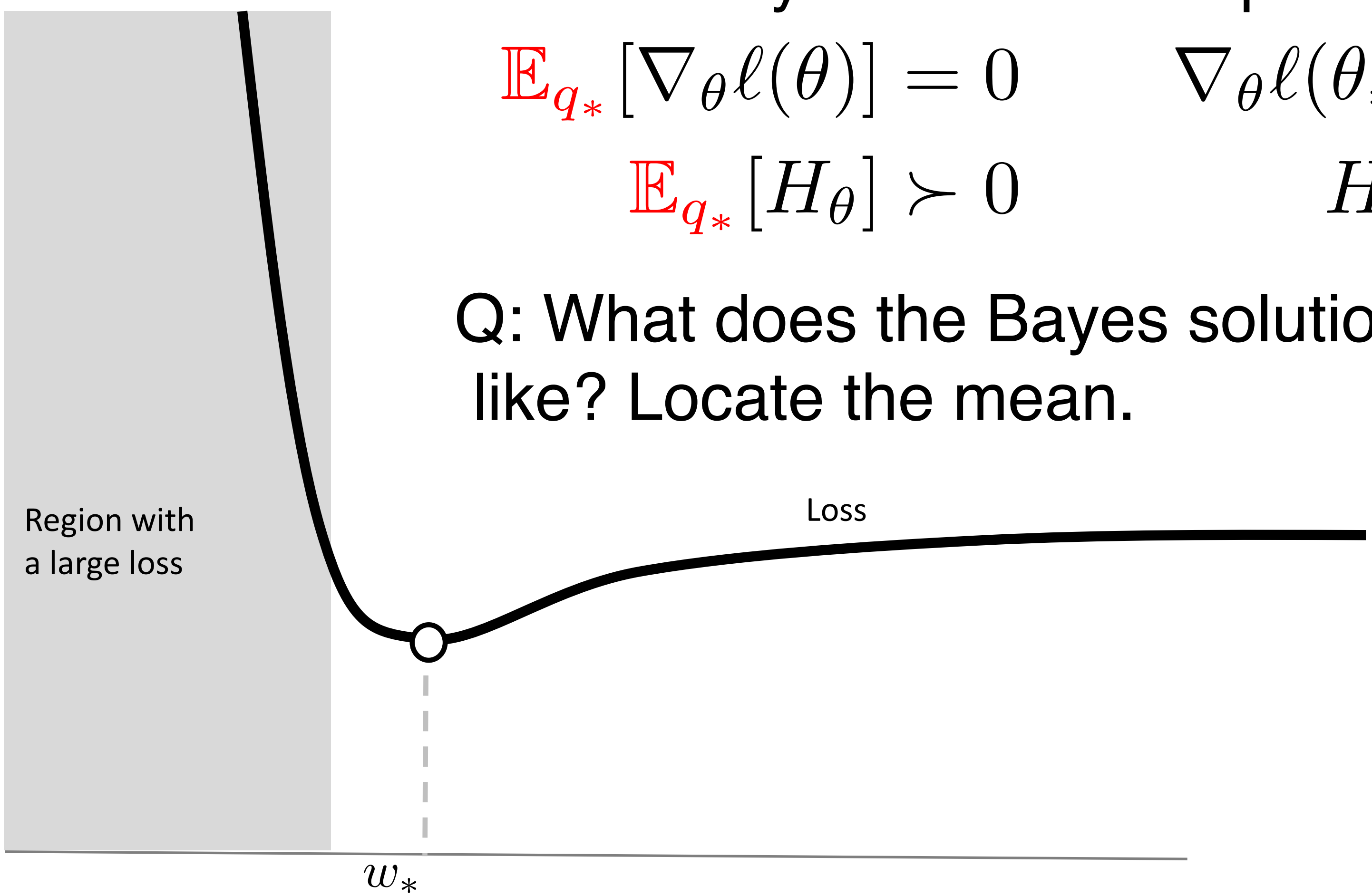
$$H_{\theta_*} \succ 0$$



The Optimization Solution

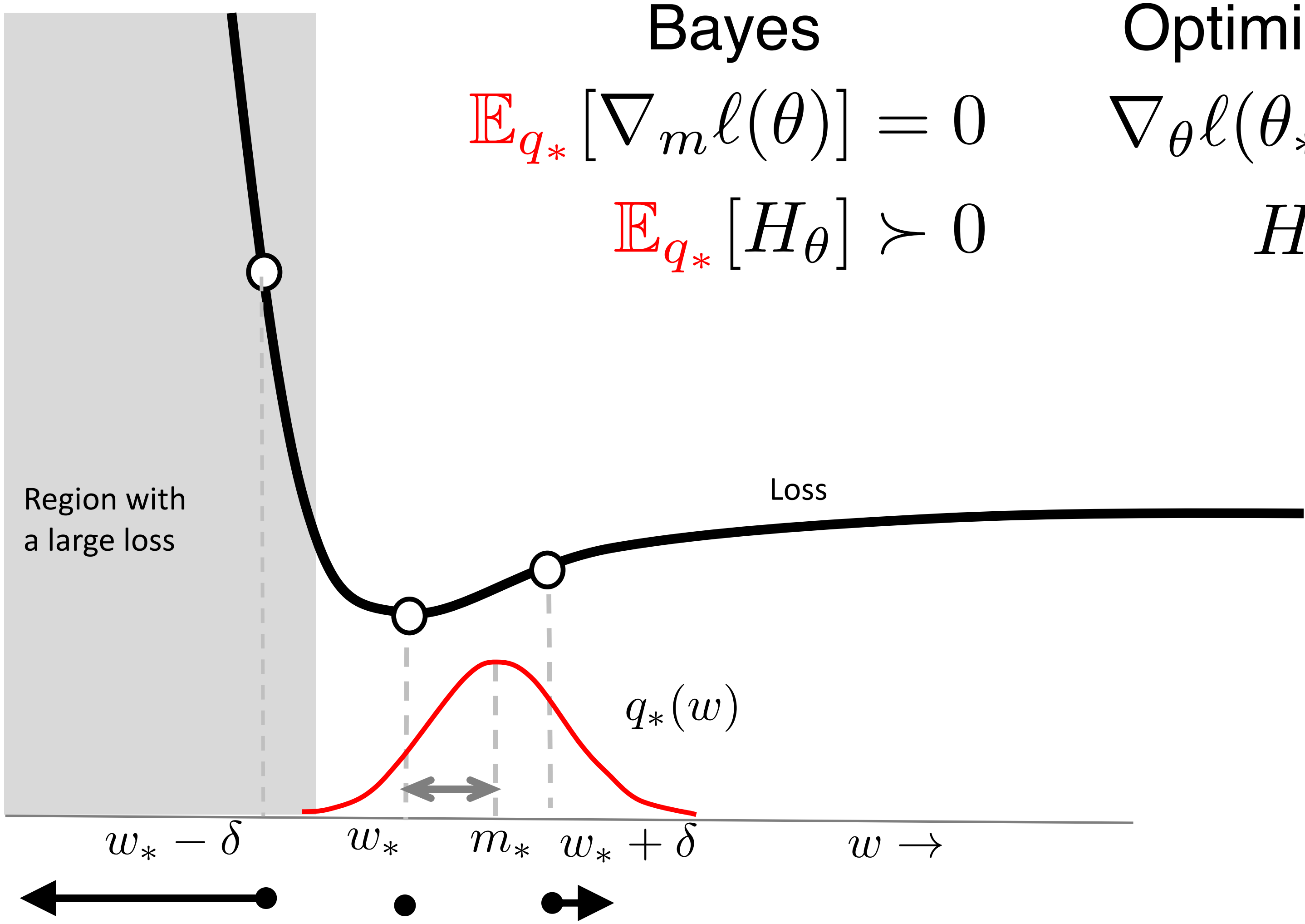
Bayes	Optimization
$\mathbb{E}_{q_*}[\nabla_\theta \ell(\theta)] = 0$	$\nabla_\theta \ell(\theta_*)] = 0$
$\mathbb{E}_{q_*}[H_\theta] \succ 0$	$H_{\theta_*} \succ 0$

Q: What does the Bayes solution look like? Locate the mean.



The Bayesian Solution

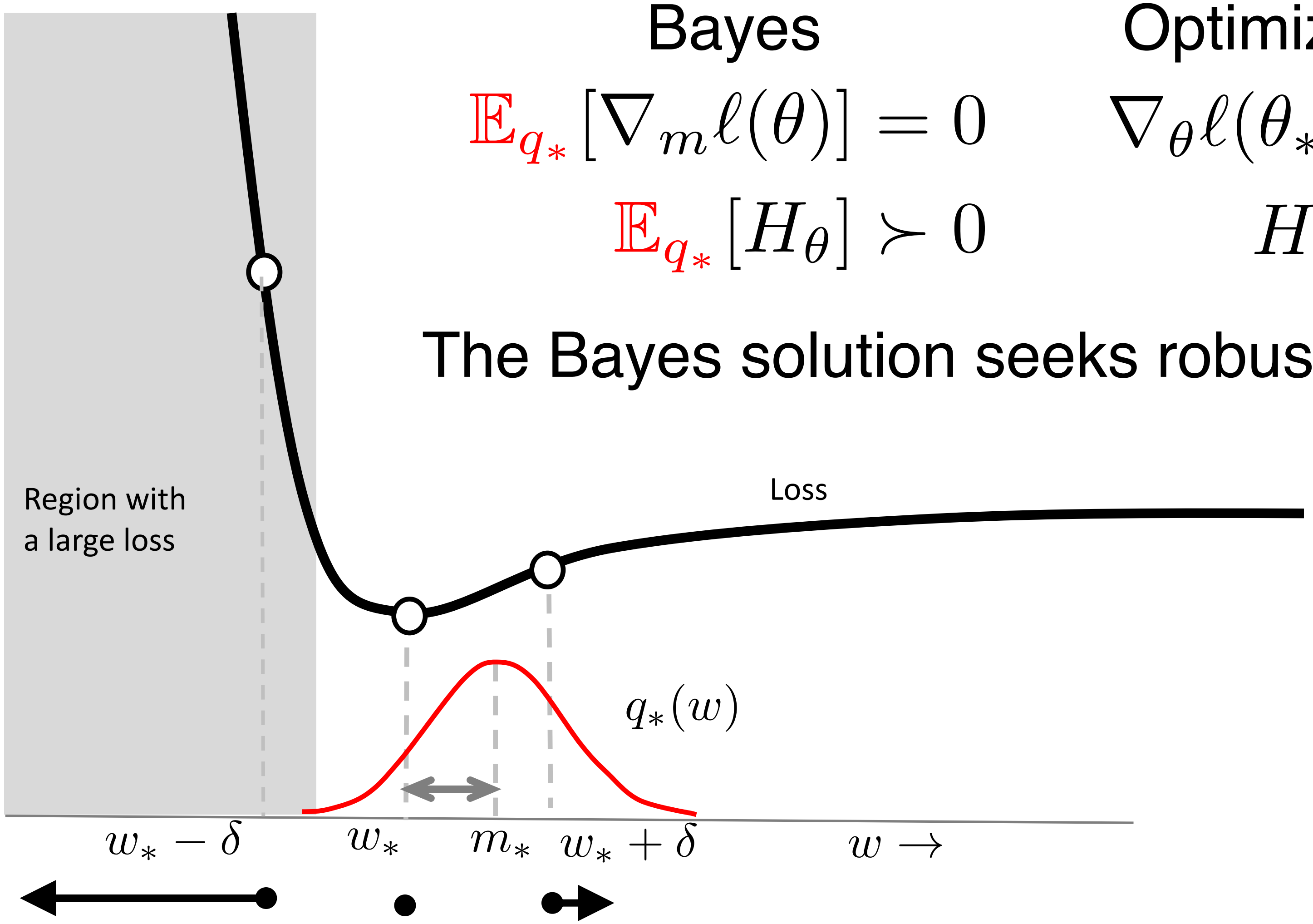
Bayes	Optimization
$\mathbb{E}_{q_*}[\nabla_m \ell(\theta)] = 0$	$\nabla_\theta \ell(\theta_*)] = 0$
$\mathbb{E}_{q_*}[H_\theta] \succ 0$	$H_{\theta_*} \succ 0$



The Bayesian Solution

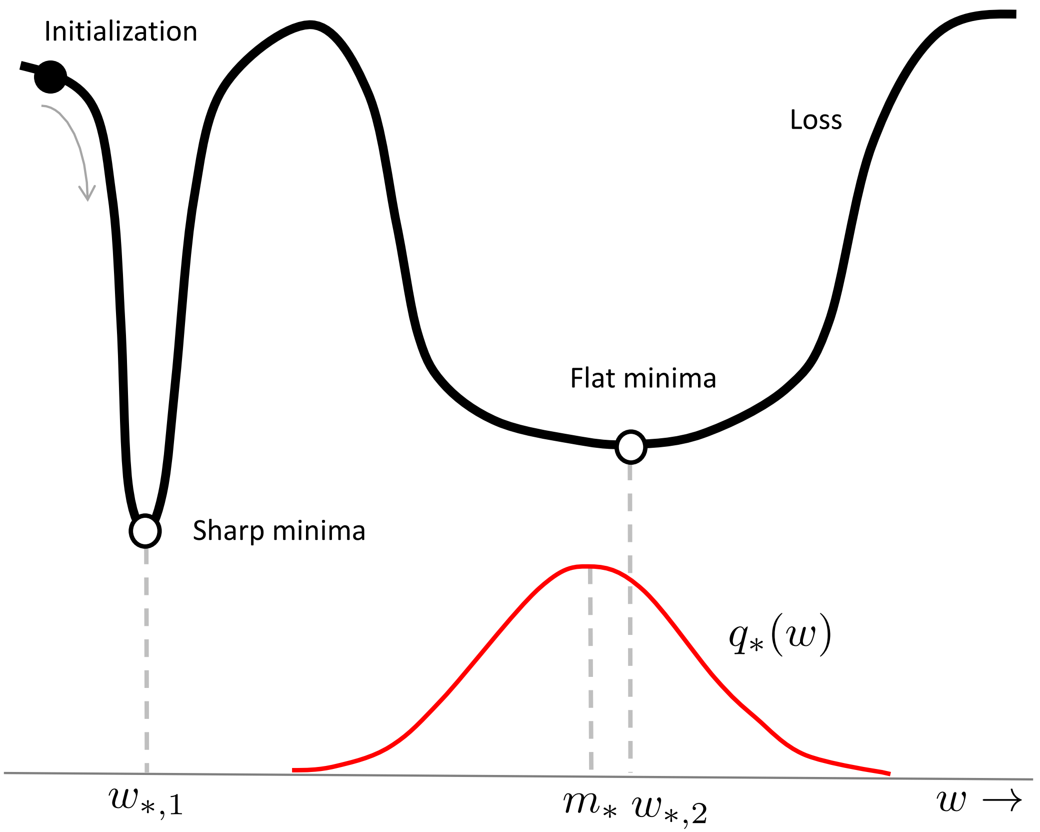
Bayes	Optimization
$\mathbb{E}_{q_*}[\nabla_m \ell(\theta)] = 0$	$\nabla_\theta \ell(\theta_*)] = 0$
$\mathbb{E}_{q_*}[H_\theta] \succ 0$	$H_{\theta_*} \succ 0$

The Bayes solution seeks robustness!



Robustness of Bayes: Example II

Bayesian solution seeks "flatter" minima



1. Khan, et al. "Variational Adaptive Newton Method for Explorative Learning" *arXiv* (2017).

RMSprop/Adam from Bayes

RMSprop

$$s \leftarrow (1-\rho)s + \rho[\hat{\nabla}\ell(\theta)]^2$$
$$\theta \leftarrow \theta - \alpha(\sqrt{s}+\delta)^{-1}\hat{\nabla}\ell(\theta)$$

Bayesian Learning rule for multivariate Gaussian

$$S \leftarrow (1-\rho)S + \rho(H_\theta)$$
$$m \leftarrow m - \alpha S^{-1}\nabla_\theta \ell(\theta)$$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).
2. Khan and Rue. “Learning-Algorithms from Bayesian Principles” (2020) (work in progress, an early draft available at https://emtiyaz.github.io/papers/learning_from_bayes.pdf)

RMSprop/Adam from Bayes

RMSprop

$$s \leftarrow (1-\rho)s + \rho[\hat{\nabla}\ell(\theta)]^2$$
$$\theta \leftarrow \theta - \alpha(\sqrt{s}+\delta)^{-1}\hat{\nabla}\ell(\theta)$$

Bayesian Learning rule for multivariate Gaussian

$$S \leftarrow (1-\rho)S + \rho(H_\theta)$$
$$m \leftarrow m - \alpha S^{-1}\nabla_\theta \ell(\theta)$$

To get RMSprop, make the following choices

- Choose Gaussian with diagonal covariance
- Replace Hessian by square of gradients
- Add square root for scaling vector

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).
2. Khan and Rue. “Learning-Algorithms from Bayesian Principles” (2020) (work in progress, an early draft available at https://emtiyaz.github.io/papers/learning_from_bayes.pdf)

RMSprop/Adam from Bayes

RMSprop

$$s \leftarrow (1-\rho)s + \rho[\hat{\nabla}\ell(\theta)]^2$$
$$\theta \leftarrow \theta - \alpha(\sqrt{s}+\delta)^{-1}\hat{\nabla}\ell(\theta)$$

Bayesian Learning rule for multivariate Gaussian

$$S \leftarrow (1-\rho)S + \rho(H_\theta)$$
$$m \leftarrow m - \alpha S^{-1}\nabla_\theta \ell(\theta)$$

To get RMSprop, make the following choices

- Choose Gaussian with diagonal covariance
- Replace Hessian by square of gradients
- Add square root for scaling vector

For Adam, use a Heavy-ball term with KL divergence as the momentum (Appendix E in [1], [2])

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).
2. Khan and Rue. “Learning-Algorithms from Bayesian Principles” (2020) (work in progress, an early draft available at https://emtiyaz.github.io/papers/learning_from_bayes.pdf)

Summary

- Gradient descent is derived using a Gaussian with fixed covariance, and estimating the mean
- Newton's method is derived using multivariate Gaussian
- RMSprop is derived using diagonal covariance
- Adam is derived by adding heavy-ball momentum term
- Dropout is derived using "spike and slab mixture".
- For "ensemble of Newton", use Mixture of Gaussians [1]
- STE is derived using Bernoulli distribution for Binary NN [2]
- To derive DL algorithms, we need to switch from a "global" to "local" approximation $\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$
- Then, to improve DL algorithms, we just need to add some "global" touch to the DL algorithms

1. Lin, Khan, Schmidt. "Fast and Simple Natural-Gradient Variational Inference with Mixture of Exponential-family Approximations." *ICML* (2019).
2. Meng, Bachman, Khan, Training Binary Neural Networks using the Bayesian Learning Rule *ICML* (2019).

Deep Learning with Bayesian Principles

- Bayesian principles as common principles
  - By computing “posterior approximations”
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and life-long learning
- Impact: Many learning-algorithms with a common set of principles.

Bayes as Optimization

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

Bayes as Optimization

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

$$\boxed{\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)}$$

Bayes as Optimization

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

$$\boxed{\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)}$$

$$= \arg\min_{q \in \mathcal{P}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

All distribution

Distribution

Entropy

Bayes as Optimization

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

$$\boxed{\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)}$$

$$= \underset{q\in\mathcal{P}}{\arg\min}\ \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

All distribution

Distribution

Entropy

$$\boxed{= \mathbb{E}_q[\ell(\theta)] + \mathbb{E}_q[\log q(\theta)]}$$

Bayes as Optimization

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

$$\boxed{\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)}$$

$$= \arg\min_{q \in \mathcal{P}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

All distribution

Distribution

Entropy

$$= \mathbb{E}_q[\ell(\theta)] + \mathbb{E}_q[\log q(\theta)] \qquad = \mathbb{E}_q\left[\log \frac{q(\theta)}{e^{-\ell(\theta)}}\right]$$

Bayes as Optimization

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

$$\boxed{\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)}$$

$$= \underset{q\in\mathcal{P}}{\arg\min}\ \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

All distribution

Distribution

Entropy

$$= \mathbb{E}_q[\ell(\theta)] + \mathbb{E}_q[\log q(\theta)] \quad = \mathbb{E}_q\left[\log \frac{q(\theta)}{e^{-\ell(\theta)}}\right]$$

$$\implies q_*(\theta) \propto e^{-\ell(\theta)}$$

Bayes as Optimization

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

$$\boxed{\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)}$$

$$= \underset{q\in\mathcal{P}}{\arg\min}\ \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

All distribution

Distribution

Entropy

$$= \mathbb{E}_q[\ell(\theta)] + \mathbb{E}_q[\log q(\theta)] \qquad = \mathbb{E}_q\left[\log \frac{q(\theta)}{e^{-\ell(\theta)}}\right]$$

$$\implies q_*(\theta) \propto e^{-\ell(\theta)} \propto p(\mathcal{D}|\theta)p(\theta)$$

Bayes as Optimization

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

$$\boxed{\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)}$$

$$= \underset{q\in\mathcal{P}}{\arg\min}\ \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

All distribution

Distribution

Entropy

$$= \mathbb{E}_q[\ell(\theta)] + \mathbb{E}_q[\log q(\theta)] \quad = \mathbb{E}_q\left[\log \frac{q(\theta)}{e^{-\ell(\theta)}}\right]$$

$$\Longrightarrow \ q_*(\theta) \propto e^{-\ell(\theta)} \propto p(\mathcal{D}|\theta)p(\theta) \ \propto p(\theta|\mathcal{D})$$

Bayes as Optimization

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

$$\boxed{\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)}$$

$$= \underset{q \in \mathcal{P}}{\arg\min}\ \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

All distribution

Distribution

Entropy

$$= \mathbb{E}_q[\ell(\theta)] + \mathbb{E}_q[\log q(\theta)] \quad = \mathbb{E}_q\left[\log \frac{q(\theta)}{e^{-\ell(\theta)}}\right]$$

$$\Longrightarrow\ q_*(\theta) \propto e^{-\ell(\theta)} \propto p(\mathcal{D}|\theta)p(\theta) \propto p(\theta|\mathcal{D})$$

Good news: This holds for a generic loss function!

Zellner (1988), Bissiri, et al. (2016), Shawe-Taylor and Williamson (1997), Cesa-Bianchi and Lugosi (2006)

References for Bayes as Optimization

$$\arg\min_{q \in \mathcal{P}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

References for Bayes as Optimization

$$\arg\min_{q \in \mathcal{P}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

- Bayesian statistics
  1. Jaynes, Edwin T. "Information theory and statistical mechanics." *Physical review* (1957)
  2. Zellner, A. "Optimal information processing and Bayes's theorem." *The American Statistician* (1988)
  3. Bissiri, Pier Giovanni, Chris C. Holmes, and Stephen G. Walker. "A general framework for updating belief distributions." *RSS: Series B (Statistical Methodology)* (2016)

References for Bayes as Optimization

$$\arg\min_{q \in \mathcal{P}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

- Bayesian statistics
  1. Jaynes, Edwin T. "Information theory and statistical mechanics." *Physical review* (1957)
  2. Zellner, A. "Optimal information processing and Bayes's theorem." *The American Statistician* (1988)
  3. Bissiri, Pier Giovanni, Chris C. Holmes, and Stephen G. Walker. "A general framework for updating belief distributions." *RSS: Series B (Statistical Methodology)* (2016)
- PAC-Bayes
  4. Shawe-Taylor, John, and Robert C. Williamson. "A PAC analysis of a Bayesian estimator." COLT 1997.
  5. Alquier, Pierre. "PAC-Bayesian bounds for randomized empirical risk minimizers." *Mathematical Methods of Statistics* 17.4 (2008): 279-304.

References for Bayes as Optimization

$$\arg\min_{q \in \mathcal{P}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

- Bayesian statistics
  1. Jaynes, Edwin T. "Information theory and statistical mechanics." *Physical review* (1957)
  2. Zellner, A. "Optimal information processing and Bayes's theorem." *The American Statistician* (1988)
  3. Bissiri, Pier Giovanni, Chris C. Holmes, and Stephen G. Walker. "A general framework for updating belief distributions." *RSS: Series B (Statistical Methodology)* (2016)
- PAC-Bayes
  4. Shawe-Taylor, John, and Robert C. Williamson. "A PAC analysis of a Bayesian estimator." COLT 1997.
  5. Alquier, Pierre. "PAC-Bayesian bounds for randomized empirical risk minimizers." *Mathematical Methods of Statistics* 17.4 (2008): 279-304.
- Online-learning (Exponential Weight Aggregate)
  6. Cesa-Bianchi, Nicolo, and Gabor Lugosi. *Prediction, learning, and games*. 2006.

References for Bayes as Optimization

$$\arg\min_{q \in \mathcal{P}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

- Bayesian statistics
  1. Jaynes, Edwin T. "Information theory and statistical mechanics." *Physical review* (1957)
  2. Zellner, A. "Optimal information processing and Bayes's theorem." *The American Statistician* (1988)
  3. Bissiri, Pier Giovanni, Chris C. Holmes, and Stephen G. Walker. "A general framework for updating belief distributions." *RSS: Series B (Statistical Methodology)* (2016)
- PAC-Bayes
  4. Shawe-Taylor, John, and Robert C. Williamson. "A PAC analysis of a Bayesian estimator." COLT 1997.
  5. Alquier, Pierre. "PAC-Bayesian bounds for randomized empirical risk minimizers." *Mathematical Methods of Statistics* 17.4 (2008): 279-304.
- Online-learning (Exponential Weight Aggregate)
  6. Cesa-Bianchi, Nicolo, and Gabor Lugosi. *Prediction, learning, and games*. 2006.
- Free-energy principle
  7. Friston, K. "The free-energy principle: a unified brain theory?." *Nature neuroscience* (2010)

Bayes with Approximate Posterior

$$\arg\min_{q \in \mathcal{P}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

All distribution

Distribution

Entropy

Restrict the set of distribution from P to Q

$$\arg\min_{q \in \mathcal{Q}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

Bayes with Approximate Posterior

$$\arg\min_{q \in \mathcal{P}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

All distribution Distribution Entropy

Restrict the set of distribution from P to Q

$$\arg\min_{q \in \mathcal{Q}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

This is known as Variational Inference, but along with the Bayesian learning rule, it enables us to derive many more algorithms (including Bayes' rule). So this is not just a method, but a principle.

Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)$$

Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} T(\theta)$$

Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top}\boxed{T(\theta)}$$

Sufficient statistics of q

Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top}\boxed{T(\theta)}$$

Sufficient statistics of q

$$\boxed{\ell(\theta) := (y - X\theta)^{\top}(y - X\theta) + \gamma\theta^{\top}\theta}$$

Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^\top \boxed{T(\theta)}$$

Sufficient statistics of q

$$\boxed{\begin{aligned}\ell(\theta) &:= (y - X\theta)^\top (y - X\theta) + \gamma\theta^\top\theta \\ &= -2\theta^\top (X^\top y) + \operatorname{Tr}\left[\theta\theta^\top \left(X^\top X + \gamma I\right)\right] + \text{cnst}\end{aligned}}$$

Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} \boxed{T(\theta)} \longleftarrow$$

Sufficient statistics of q

$$\boxed{\begin{aligned}\ell(\theta) &:= (y - X\theta)^{\top}(y - X\theta) + \gamma\theta^{\top}\theta \\ &= -2\theta^{\top}(X^{\top}y) + \mathrm{Tr}\left[\theta\theta^{\top}\left(X^{\top}X + \gamma I\right)\right] + \mathrm{cnst}\end{aligned}}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}\mu$$

Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^\top \boxed{T(\theta)}$$

Sufficient statistics of q

$$\boxed{\begin{aligned}\ell(\theta) &:= (y - X\theta)^\top (y - X\theta) + \gamma\theta^\top\theta \\ &= -2\theta^\top (X^\top y) + \mathrm{Tr}\left[\theta\theta^\top \left(X^\top X + \gamma I\right)\right] + \mathrm{cnst}\end{aligned}}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}\mu \implies \nabla_\mu \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}$$

Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} \boxed{T(\theta)}$$

Sufficient statistics of q

$$\boxed{\begin{aligned}\ell(\theta) &:= (y - X\theta)^{\top}(y - X\theta) + \gamma\theta^{\top}\theta \\ &= -2\theta^{\top}(X^{\top}y) + \mathrm{Tr}\left[\theta\theta^{\top}\left(X^{\top}X + \gamma I\right)\right] + \mathrm{cnst}\end{aligned}}$$

$$\Longrightarrow \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}\mu \quad \Longrightarrow \nabla_{\mu}\mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}$$

$$\lambda \leftarrow \lambda - \rho\nabla_{\mu}\left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q)\right)$$

Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} \boxed{T(\theta)}$$

Sufficient statistics of q

$$\boxed{\begin{aligned}\ell(\theta) &:= (y - X\theta)^{\top}(y - X\theta) + \gamma\theta^{\top}\theta \\ &= -2\theta^{\top}(X^{\top}y) + \mathrm{Tr}\left[\theta\theta^{\top}\left(X^{\top}X + \gamma I\right)\right] + \mathrm{cnst}\end{aligned}}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}\mu \implies \nabla_{\mu}\mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}$$

$$\lambda \leftarrow \lambda - \rho\left(\nabla_{\mu}\mathbb{E}_q[\ell(\theta)] + \lambda\right)$$

Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^\top \boxed{T(\theta)}$$

Sufficient statistics of q

$$\boxed{\begin{aligned}\ell(\theta) &:= (y - X\theta)^\top (y - X\theta) + \gamma\theta^\top\theta \\ &= -2\theta^\top (X^\top y) + \operatorname{Tr}\left[\theta\theta^\top \left(X^\top X + \gamma I\right)\right] + \text{cnst}\end{aligned}}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}\mu \implies \nabla_\mu \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}$$

$$\lambda \leftarrow \lambda - \rho\left(-\lambda_{\mathcal{D}} + \lambda\right)$$

Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^\top \boxed{T(\theta)}$$

Sufficient statistics of q

$$\boxed{\begin{aligned}\ell(\theta) &:= (y - X\theta)^\top (y - X\theta) + \gamma\theta^\top\theta \\ &= -2\theta^\top (X^\top y) + \operatorname{Tr}\left[\theta\theta^\top \left(X^\top X + \gamma I\right)\right] + \text{cnst}\end{aligned}}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}\mu \implies \nabla_\mu \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}$$

$$\lambda \leftarrow \lambda - \rho\left(-\lambda_{\mathcal{D}} + \lambda\right) \qquad \implies \lambda_* = \lambda_{\mathcal{D}}$$

Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^\top \boxed{T(\theta)}$$

Sufficient statistics of q

$$\boxed{\begin{aligned}\ell(\theta) &:= (y - X\theta)^\top (y - X\theta) + \gamma\theta^\top\theta \\ &= -2\theta^\top (X^\top y) + \mathrm{Tr}\left[\theta\theta^\top \left(X^\top X + \gamma I\right)\right] + \mathrm{cnst}\end{aligned}}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}\mu \implies \nabla_\mu \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}$$

$$\lambda \leftarrow \lambda - \rho\left(-\lambda_{\mathcal{D}} + \lambda\right) \qquad \implies \lambda_* = \lambda_{\mathcal{D}}$$

$$\boxed{S_* = X^\top X + \gamma I}$$

Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} \boxed{T(\theta)}$$ ← Sufficient statistics of q

$$\boxed{\begin{aligned} \ell(\theta) &:= (y - X\theta)^{\top}(y - X\theta) + \gamma\theta^{\top}\theta \\ &= -2\theta^{\top}(X^{\top}y) + \operatorname{Tr}\left[\theta\theta^{\top}\left(X^{\top}X + \gamma I\right)\right] + \text{cnst} \end{aligned}}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}\mu \quad \implies \nabla_{\mu}\mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}$$

$$\lambda \leftarrow \lambda - \rho\left(-\lambda_{\mathcal{D}} + \lambda\right) \qquad \implies \lambda_* = \lambda_{\mathcal{D}}$$

$$\boxed{S_* = X^{\top}X + \gamma I \qquad m_* = (X^{\top}X + \gamma I)^{-1}X^{\top}y}$$

Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).

Conjugate Bayesian Inference from Bayesian Principles

The following algorithms can be obtained by setting $\lambda_* = \lambda_{\mathcal{D}}$

- Forward-backward algorithm [2]
  - Kalman filters, HMM etc.
- Stochastic Variational Inference [3]
- Variational message passing [4]

1. Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models.” AIstats (2017).
2. Binder et al.. Space-Efficient Inference in Dynamic Probabilistic Networks. IJCAI (1997).
3. Hoffman et al. Stochastic variational inference. JMLR (2013)
4. Winn and Bishop. "Variational message passing." JMLR (2005)

Laplace Approximation

Derived by choosing a multivariate Gaussian, then running the following Newton's update

$$\boxed{\begin{aligned} m &\leftarrow m - \rho S^{-1} \nabla_m \ell(m) \\ S &\leftarrow (1-\rho) S + \rho H_m \end{aligned}}$$

Hessian at *m*

Laplace Approximation

Derived by choosing a multivariate Gaussian, then running the following Newton's update

$$\boxed{\begin{aligned} m &\leftarrow m - \rho S^{-1} \nabla_m \ell(m) \\ S &\leftarrow (1-\rho) S + \rho H_m \end{aligned}}$$

Hessian at *m*

Bayesian principles we discussed are general principles to derive learning algorithms

Laplace Approximation

Derived by choosing a multivariate Gaussian, then running the following Newton's update

$$\boxed{\begin{aligned} m &\leftarrow m - \rho S^{-1} \nabla_m \ell(m) \\ S &\leftarrow (1-\rho) S + \rho H_m \end{aligned}}$$

Hessian at *m*

Bayesian principles we discussed are general principles to derive learning algorithms

Calling them variational inference limits their scope!

References for Posterior Approximations

$$\arg\min_{q \in \mathcal{Q}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

References for Posterior Approximations

$$\arg\min_{q \in \mathcal{Q}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

- Variational inference

1. Hinton, Geoffrey, and Drew Van Camp. "Keeping neural networks simple by minimizing the description length of the weights." *COLT* 1993.
2. Jordan, Michael I., et al. "An introduction to variational methods for graphical models." *Machine learning* 37.2 (1999): 183-233.

References for Posterior Approximations

$$\arg\min_{q \in \mathcal{Q}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

- Variational inference

1. Hinton, Geoffrey, and Drew Van Camp. "Keeping neural networks simple by minimizing the description length of the weights." *COLT* 1993.
2. Jordan, Michael I., et al. "An introduction to variational methods for graphical models." *Machine learning* 37.2 (1999): 183-233.

- Entropy-regularized / Maximum-entropy RL

3. Williams, Ronald J., and Jing Peng. "Function optimization using connectionist reinforcement learning algorithms." *Connection Science* 3.3 (1991): 241-268.
4. Ziebart, Brian D. Modeling purposeful adaptive behavior with the principle of maximum causal entropy. Diss. figshare, 2010. (see chapter 5)

References for Posterior Approximations

$$\arg\min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

- Variational inference

1. Hinton, Geoffrey, and Drew Van Camp. "Keeping neural networks simple by minimizing the description length of the weights." *COLT* 1993.
2. Jordan, Michael I., et al. "An introduction to variational methods for graphical models." *Machine learning* 37.2 (1999): 183-233.

- Entropy-regularized / Maximum-entropy RL

3. Williams, Ronald J., and Jing Peng. "Function optimization using connectionist reinforcement learning algorithms." *Connection Science* 3.3 (1991): 241-268.
4. Ziebart, Brian D. Modeling purposeful adaptive behavior with the principle of maximum causal entropy. Diss. figshare, 2010. (see chapter 5)

- Parameter-Space Exploration in RL

5. Rückstiess, Thomas, et al. "Exploring parameter space in reinforcement learning." *Paladyn, Journal of Behavioral Robotics* 1.1 (2010): 14-24.
6. Plappert, Matthias, et al. "Parameter space noise for exploration." *arXiv preprint arXiv:1706.01905* (2017)
7. .Fortunato, Meire, et al. "Noisy networks for exploration." *arXiv preprint arXiv:1706.10295* (2017).

More References for Posterior Approximations

- Evolution strategy $\arg\min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)}[\ell(\theta)]$
    1. Ingo Rechenberg, Evolutionsstrategie – Optimierung technischer Systeme nach Prinzipien der biologischen Evolution (PhD thesis) 1971.
- Gaussian Homotopy
    2. Mobahi, Hossein, and John W. Fisher III. "A theoretical analysis of optimization by Gaussian continuation." *Twenty-Ninth AAAI Conference on Artificial Intelligence*. 2015.
- Smoothing-based Optimization
    3. Leordeanu, Marius, and Martial Hebert. "Smoothing-based optimization." *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2008.
- Graduated Optimization
    4. Hazan, Elad, Kfir Yehuda Levy, and Shai Shalev-Shwartz. "On graduated optimization for stochastic non-convex problems." *International conference on machine learning*. 2016.
- Stochastic Search
    5. Zhou, Enlu, and Jiaqiao Hu. "Gradient-based adaptive stochastic search for non-differentiable optimization." *IEEE Transactions on Automatic Control* 59.7 (2014): 1818-1832.

Bayesian Learning Rule and Related Works

$$\min_{q \in \mathcal{Q}} \; \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

Bayes learning rule: $\lambda \leftarrow \lambda - \rho \nabla_{\mu} \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q) \right)$

Natural-Gradient VI: $\lambda \leftarrow \lambda - \rho F_q{}^{-1} \nabla_{\lambda} \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q) \right)$

Fisher Information Matrix

1. Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).
2. Raskutti, Garvesh, and Sayan Mukherjee. "The information geometry of mirror descent." *IEEE Transactions on Information Theory* 61.3 (2015): 1451-1457.

Bayesian Learning Rule and Related Works

$$\min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

Bayes learning rule: $\lambda \leftarrow \lambda - \rho \nabla_{\mu} \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q) \right)$

Natural-Gradient VI: $\lambda \leftarrow \lambda - \rho F_q^{-1} \nabla_{\lambda} \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q) \right)$

Fisher Information Matrix

Also equivalent to a mirror-descent algorithm.The Geometry of the mirror-descent is defined by the log partition function of the posterior approximation.

1. Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).
2. Raskutti, Garvesh, and Sayan Mukherjee. "The information geometry of mirror descent." *IEEE Transactions on Information Theory* 61.3 (2015): 1451-1457.

References for Step C: Natural-Gradient VI

1. Sato, Masa-aki. "Fast learning of on-line EM algorithm." Technical Report, ATR Human Information Processing Research Laboratories (1999).
2. Sato, Masa-Aki. "Online model selection based on the variational Bayes." *Neural computation* 13.7 (2001): 1649-1681.
3. Winn, John, and Christopher M. Bishop. "Variational message passing." *Journal of Machine Learning Research* 6.Apr (2005): 661-694.
4. Honkela, Antti, et al. "Approximate Riemannian conjugate gradient learning for fixed-form variational Bayes." *Journal of Machine Learning Research* 11.Nov (2010): 3235-3268.
5. Knowles, David A., and Tom Minka. "Non-conjugate variational message passing for multinomial and binary regression." *NeurIPS*. (2011).
6. Hoffman, Matthew D., et al. "Stochastic variational inference." *JMLR* (2013).
7. Salimans, Tim, and David A. Knowles. "Fixed-form variational posterior approximation through stochastic linear regression." *Bayesian Analysis* 8.4 (2013): 837-882.
8. Sheth, Rishit, and Roni Khardon. "Monte Carlo Structured SVI for Two-Level Non-Conjugate Models." *arXiv preprint arXiv:1612.03957* (2016).
9. Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." AIstats (2017).
10. Khan and Nielsen. "Fast yet simple natural-gradient descent for variational inference in complex models." (*2018) ISITA*.
11. Zhang, Guodong, et al. "Noisy natural gradient as variational inference." *ICML* (2018).

Black-Box VI & Bayesian Learning rule

Bayes learning rule: $\lambda \leftarrow \lambda - \rho \nabla_{\mu} \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q) \right)$

Black-Box VI [1]: $\lambda \leftarrow \lambda - \rho \nabla_{\lambda} \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q) \right)$

Black-box VI is more generally applicable (beyond exponential-family), but we cannot derive learning-algorithms from it (even for conjugate Bayesian models)

1. Ranganath, Rajesh, Sean Gerrish, and David Blei. "Black box variational inference." *Artificial Intelligence and Statistics*. 2014.

Learning-Algorithms from Bayesian Principles

Bayesian learning rule: $\lambda \leftarrow \lambda - \rho \nabla_{\mu} \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q) \right)$

Given a loss, we can recover a variety of learning algorithms by choosing an appropriate q

- Classical algorithms: Least-squares, gradient descent, Newton's method, Kalman filters, Baum-Welch, Forward-backward, etc.
- Bayesian inference: EM, Laplace's method, SVI, VMP.
- Deep learning: SGD, RMSprop, Adam.
- Reinforcement learning: parameter-space exploration, natural policy-search.
- Continual learning: Elastic-weight consolidation.
- Online learning: Exponential-weight average.
- Global optimization: Natural evolutionary strategies, Gaussian homotopy, continuation method & smoothed optimization.

1. Khan and Rue. "Learning-Algorithms from Bayesian Principles" (2020) (work in progress, an early draft available at https://emtiyaz.github.io/papers/learning_from_bayes.pdf)

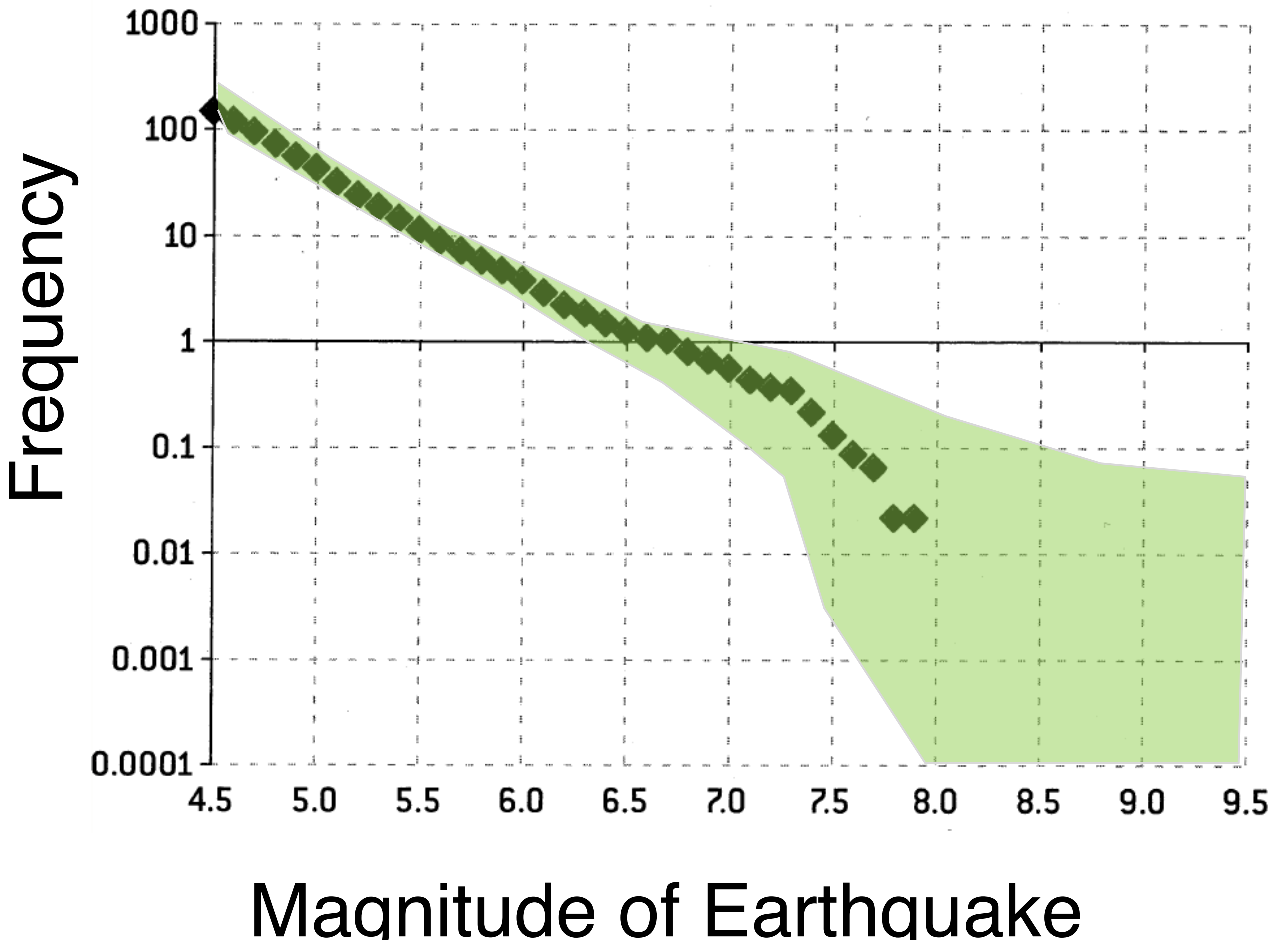
Deep Learning with Bayesian Principles

- Bayesian principles as common principles
  - By computing "posterior approximations"
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and life-long learning
- Impact: Many learning-algorithms with a common set of principles.

Uncertainty Estimation for Deep Learning

New deep-learning algorithms

Uncertainty for Robust Decisions



Uncertainty: "What the model does not know"

Choose less risky options!

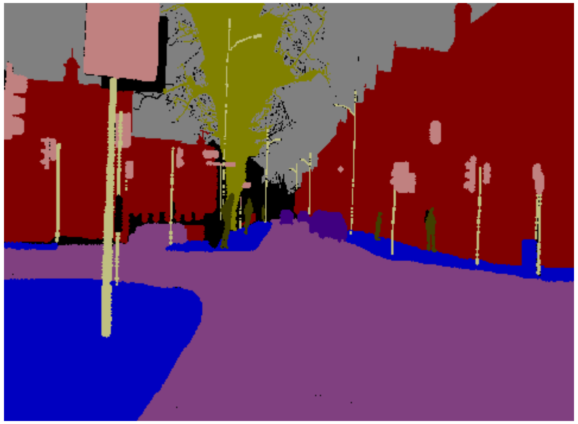
Avoid data bias with uncertainty!

Real data from Tohoku (Japan). Example taken from Nate Silver's book "The signal and noise"

Uncertainty Estimation for Image segmentation

Image

True Segments

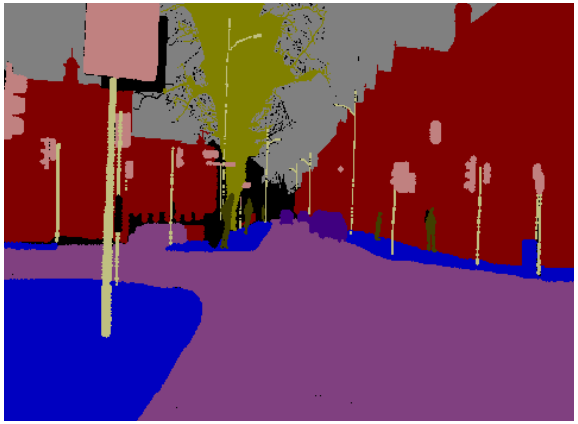


Kendall, Alex, Yarin Gal, and Roberto Cipolla. "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics." *CVPR*. 2018.

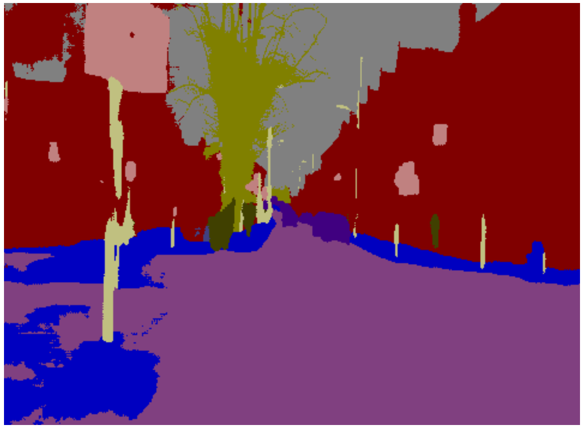
Uncertainty Estimation for Image segmentation

Image

True Segments



Prediction

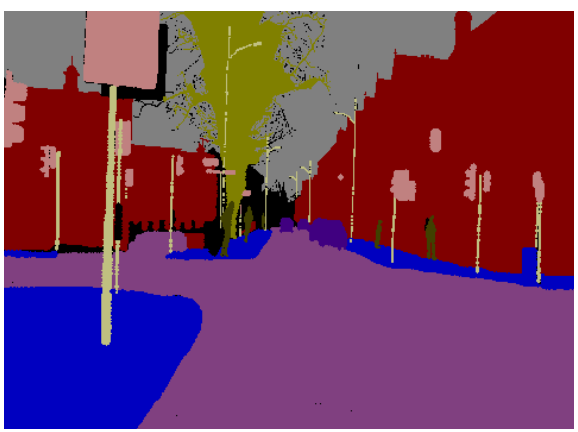


Kendall, Alex, Yarin Gal, and Roberto Cipolla. "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics." *CVPR*. 2018.

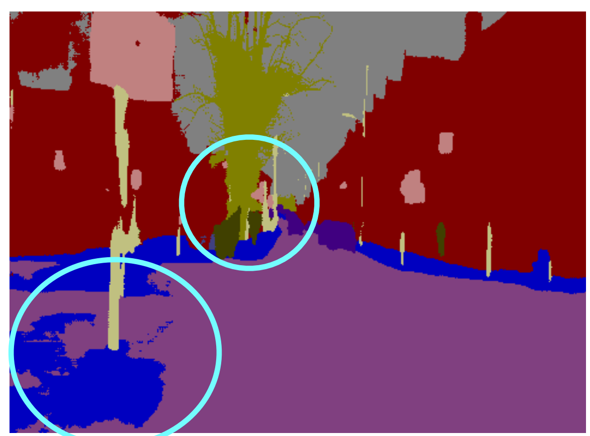
Uncertainty Estimation for Image segmentation

Image

True Segments



Prediction

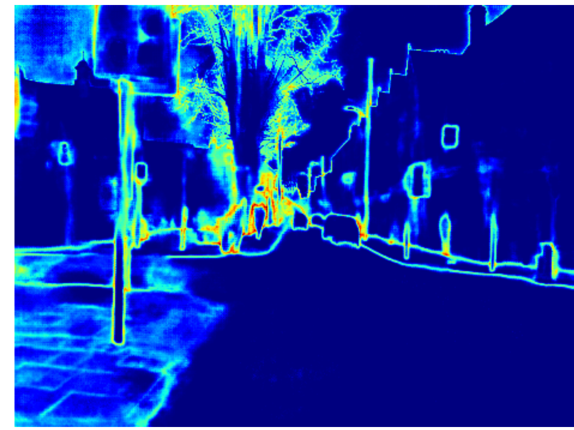


Kendall, Alex, Yarin Gal, and Roberto Cipolla. "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics." *CVPR*. 2018.

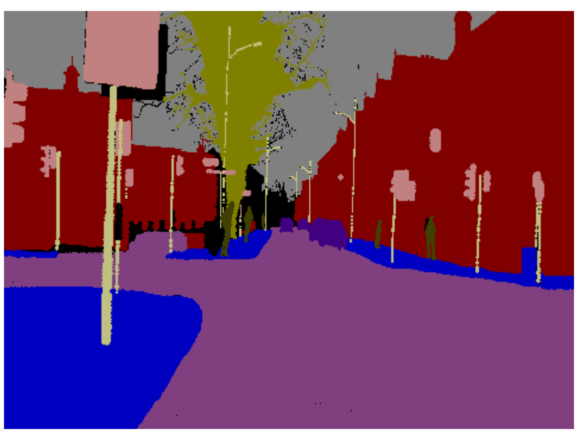
Uncertainty Estimation for Image segmentation

Image

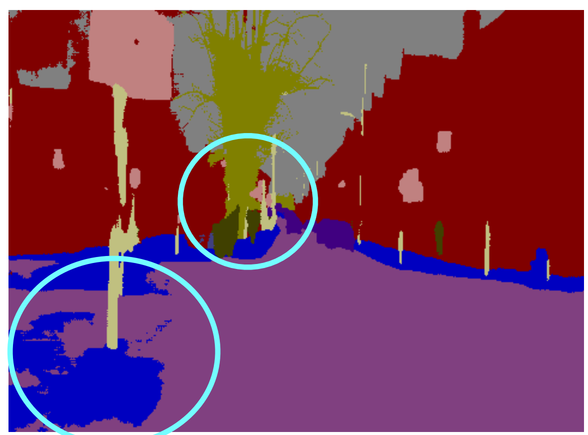
Uncertainty

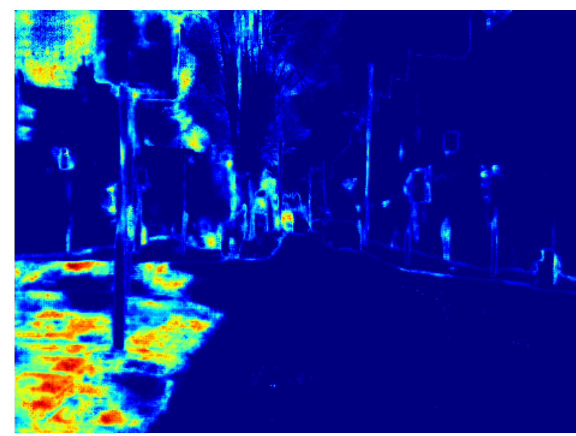


True Segments



Prediction





Kendall, Alex, Yarin Gal, and Roberto Cipolla. "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics." *CVPR*. 2018.

(Some) Bayesian Deep Learning Methods

1. Gal and Ghahramani. "Dropout as a bayesian approximation…" *ICML*. 2016.
2. Maddox, Wesley, et al. "A simple baseline for bayesian uncertainty in deep learning." *arXiv* (2019).
3. Ritter et al. "A scalable laplace approximation for neural networks." (2018).
4. Graves, Alex. "Practical variational inference for neural networks." *NeurIPS* (2011).
5. Blundell, Charles, et al. "Weight uncertainty in neural networks." *ICML* (2015).

(Some) Bayesian Deep Learning Methods

- SGD based (MC-dropout [1], SWAG [2], Laplace [3])
  - Pros: Scales well to large problems
  - Cons: Not flexible

1. Gal and Ghahramani. "Dropout as a bayesian approximation…" *ICML*. 2016.
2. Maddox, Wesley, et al. "A simple baseline for bayesian uncertainty in deep learning." *arXiv* (2019).
3. Ritter et al. "A scalable laplace approximation for neural networks." (2018).
4. Graves, Alex. "Practical variational inference for neural networks." *NeurIPS* (2011).
5. Blundell, Charles, et al. "Weight uncertainty in neural networks." *ICML* (2015).

(Some) Bayesian Deep Learning Methods

- SGD based (MC-dropout [1], SWAG [2], Laplace [3])
  - Pros: Scales well to large problems
  - Cons: Not flexible
- Variational inference methods [4,5]

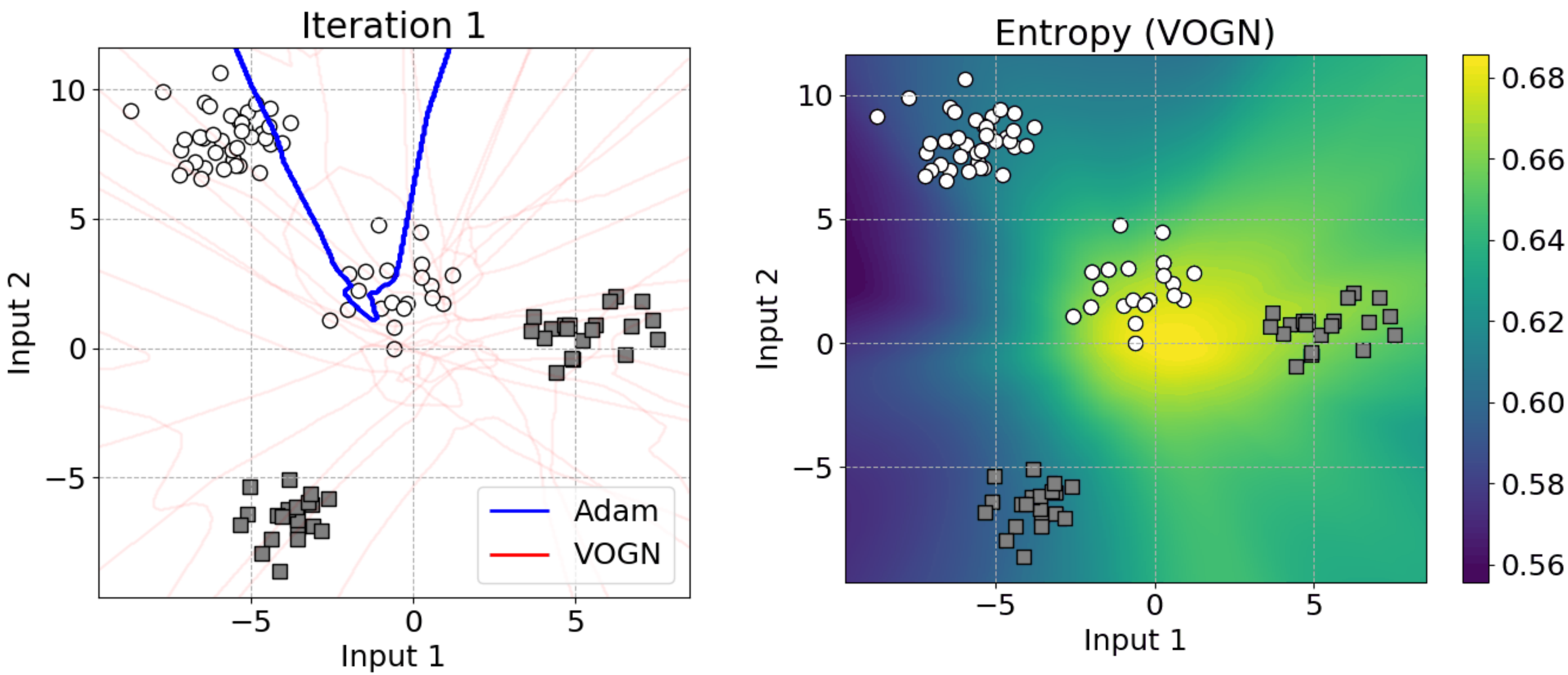
$$\lambda \leftarrow \lambda - \rho \nabla_{\lambda} \left(\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q) \right)$$

  - Pros: Enable flexible distributions
  - Cons: Do not scale to large problems (ImageNet)

1. Gal and Ghahramani. "Dropout as a bayesian approximation…" *ICML*. 2016.
2. Maddox, Wesley, et al. "A simple baseline for bayesian uncertainty in deep learning." *arXiv* (2019).
3. Ritter et al. "A scalable laplace approximation for neural networks." (2018).
4. Graves, Alex. "Practical variational inference for neural networks." *NeurIPS* (2011).
5. Blundell, Charles, et al. "Weight uncertainty in neural networks." *ICML* (2015).

Scaling up VI to ImageNet

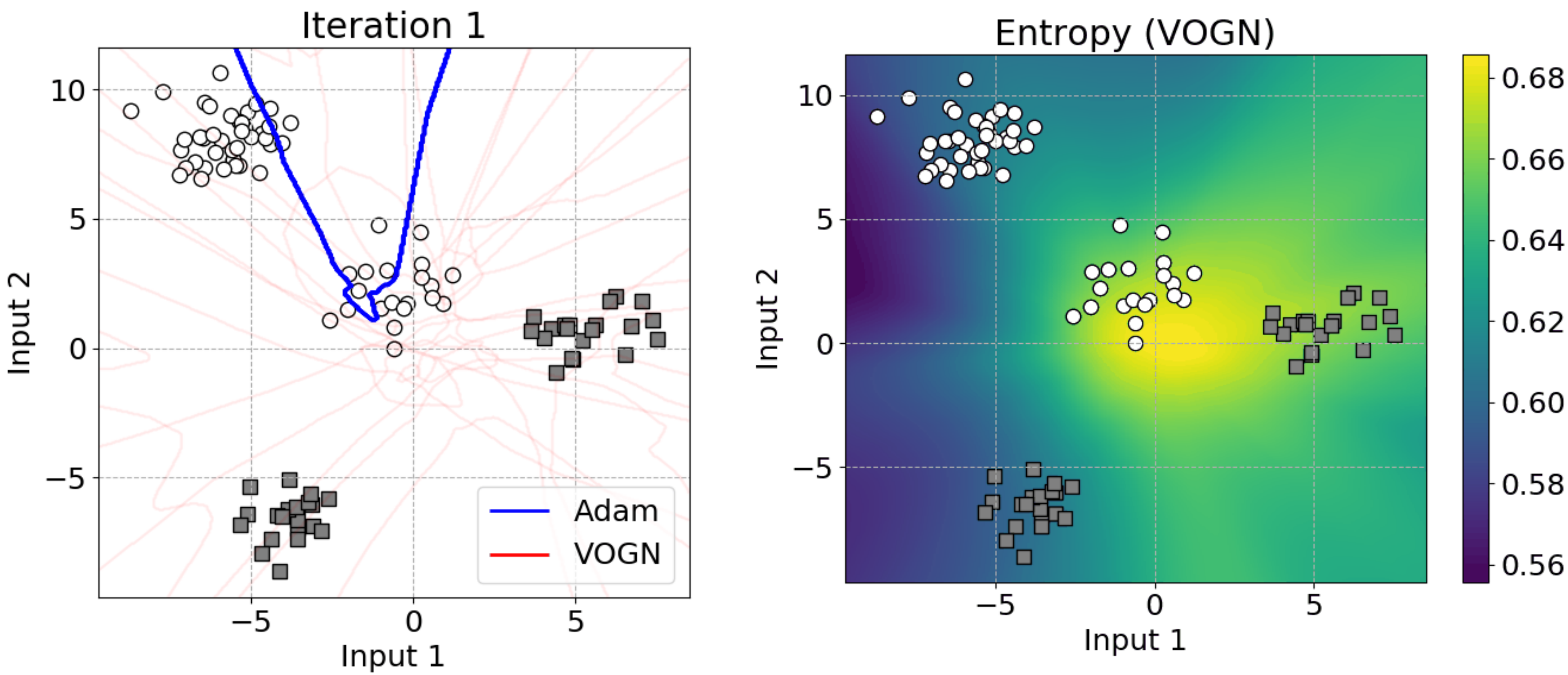
VOGN, an Adam-like algorithm, for uncertainty



1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).
2. Osawa et al. “Practical Deep Learning with Bayesian Principles.” NeurIPS (2019).

Scaling up VI to ImageNet

VOGN, an Adam-like algorithm, for uncertainty



1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).
2. Osawa et al. “Practical Deep Learning with Bayesian Principles.” NeurIPS (2019).

Variational Online Gauss-Newton

- Improve RMSprop with the Bayesian "touch"
  - Remove the "local" approximation $\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$
  - Use a second-order approximation
  - No square root of the scale
- Improve VOGN by using deep learning tricks
  - Momentum, batch norm, data augmentation etc

RMSprop

$$g \leftarrow \hat{\nabla}\ell(\theta)$$
$$s \leftarrow (1-\rho)s + \rho g^2$$
$$\theta \leftarrow \theta - \alpha(\sqrt{s}+\delta)^{-1} g$$

VOGN

$$g \leftarrow \hat{\nabla}\ell(\theta), \text{ where } \theta \sim \mathcal{N}(m, \sigma^2)$$
$$s \leftarrow (1-\rho)s + \rho(\Sigma_i g_i^2)$$
$$m \leftarrow m - \alpha(s+\gamma)^{-1}\nabla_\theta \ell(\theta)$$
$$\sigma^2 \leftarrow (s+\gamma)^{-1}$$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).
2. Osawa et al. "Practical Deep Learning with Bayesian Principles." NeurIPS (2019).

Adam to VOGN

"Adam" to "VOGN" in two lines of code change.

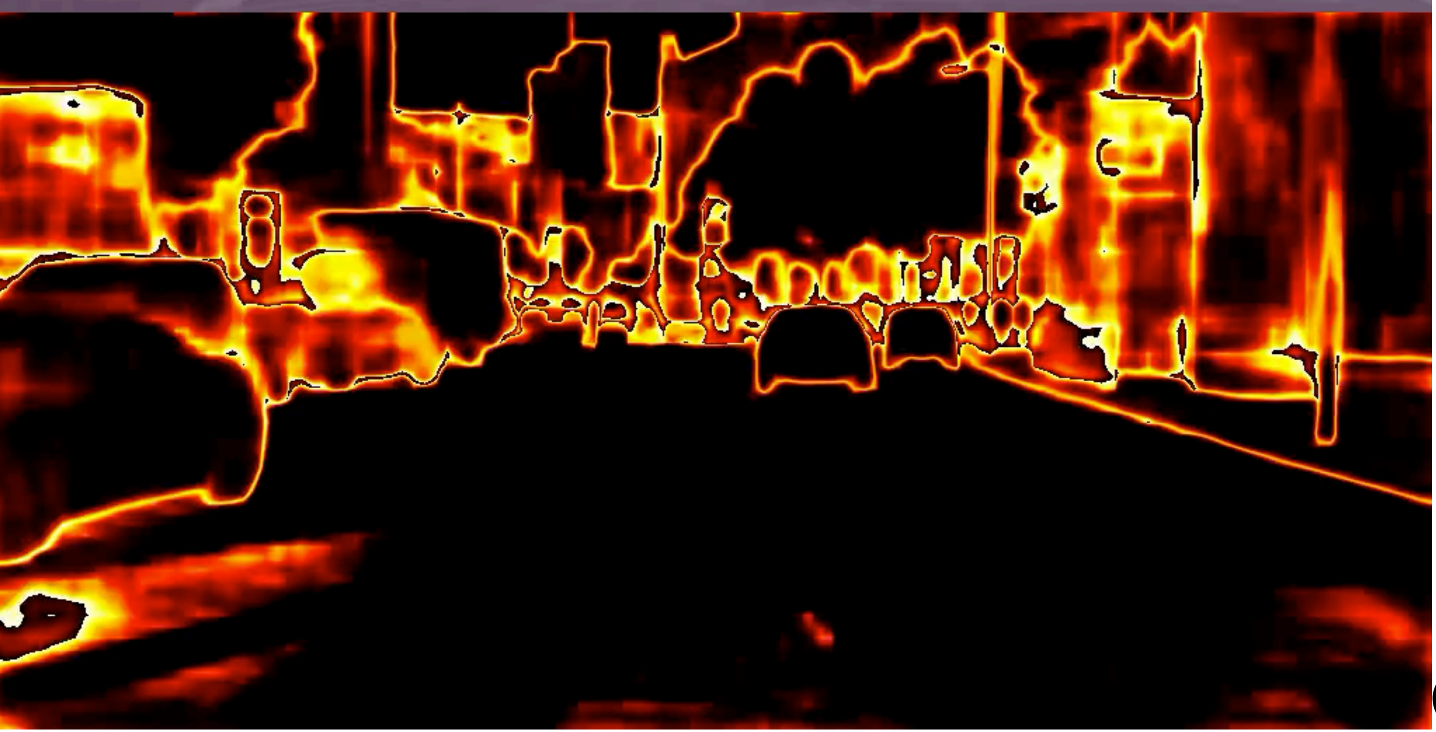
```
import torch
+import torchsso

train_loader = torch.utils.data.DataLoader(train_dataset)
model = MLP()

-optimizer = torch.optim.Adam(model.parameters())
+optimizer = torchsso.optim.VOGN(model, dataset_size=len(train_loader.dataset))
```

Available at https://github.com/team-approx-bayes/dl-with-bayes

Uses many practical tricks of DL to scale Bayes

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).
2. Osawa et al. "Practical Deep Learning with Bayesian Principles." NeurIPS (2019).

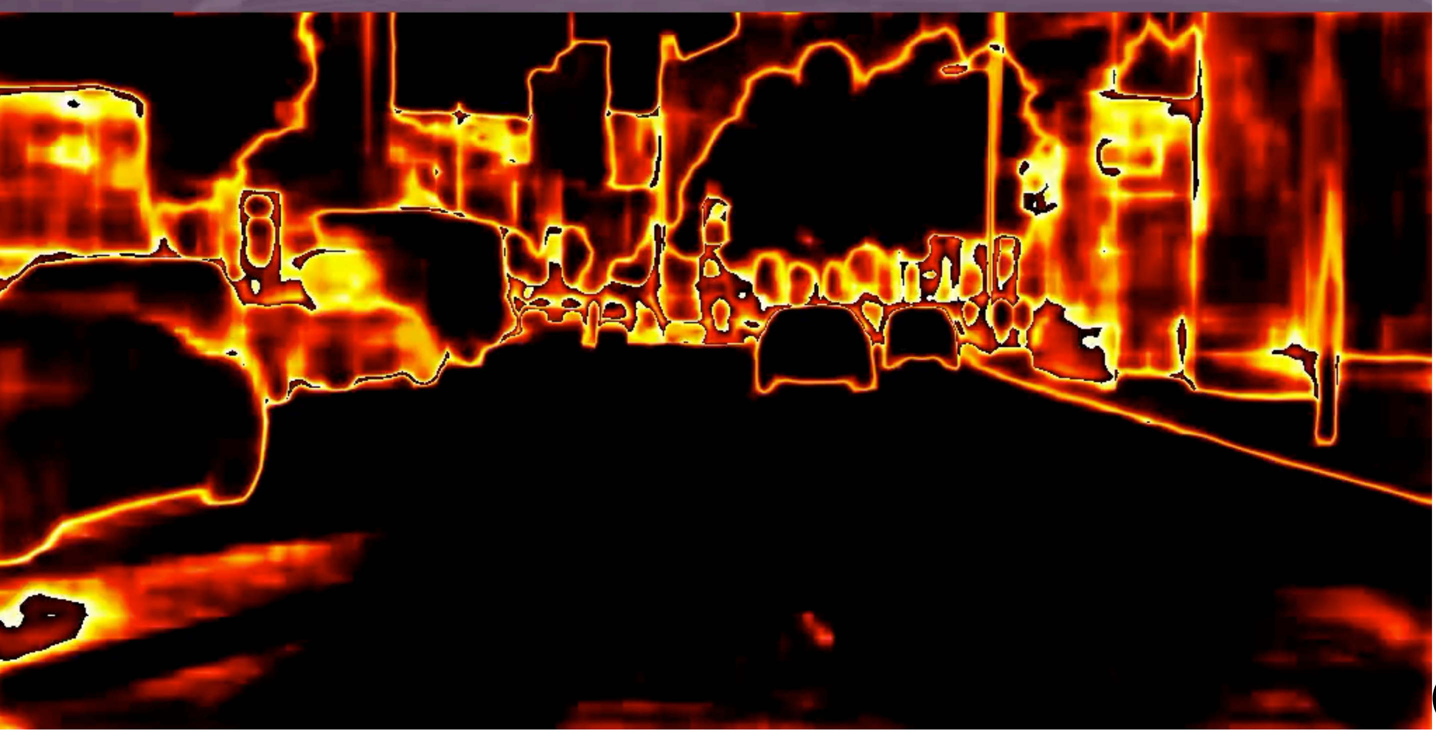
Image Segmentation



Uncertainty (entropy of class probs)

(By Roman Bachmann)

Image Segmentation

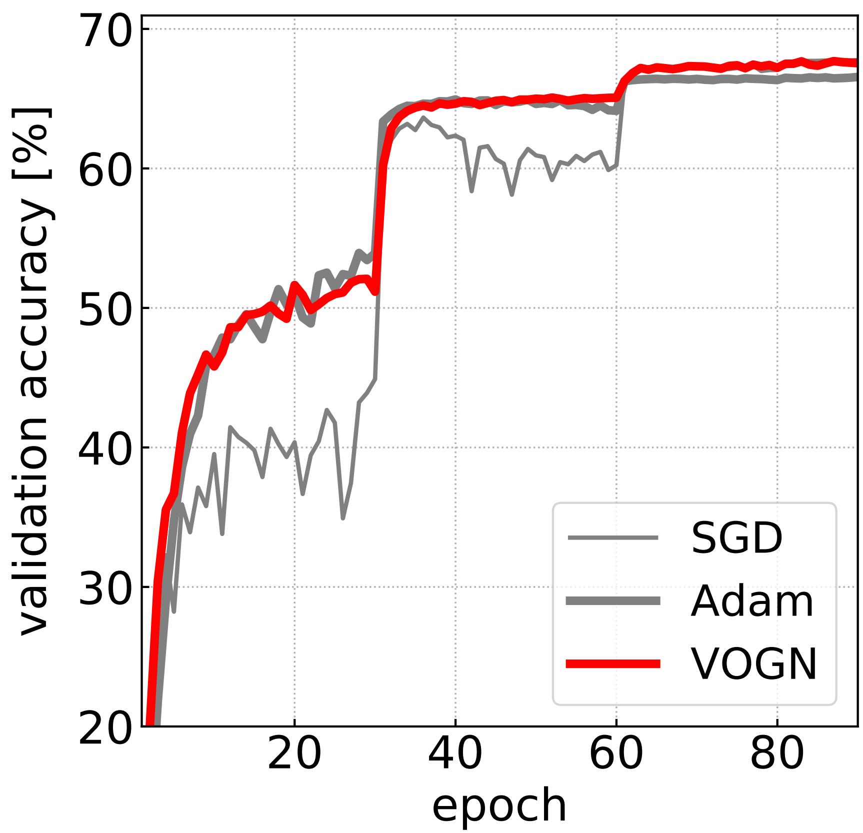


Uncertainty (entropy of class probs)

(By Roman Bachmann)

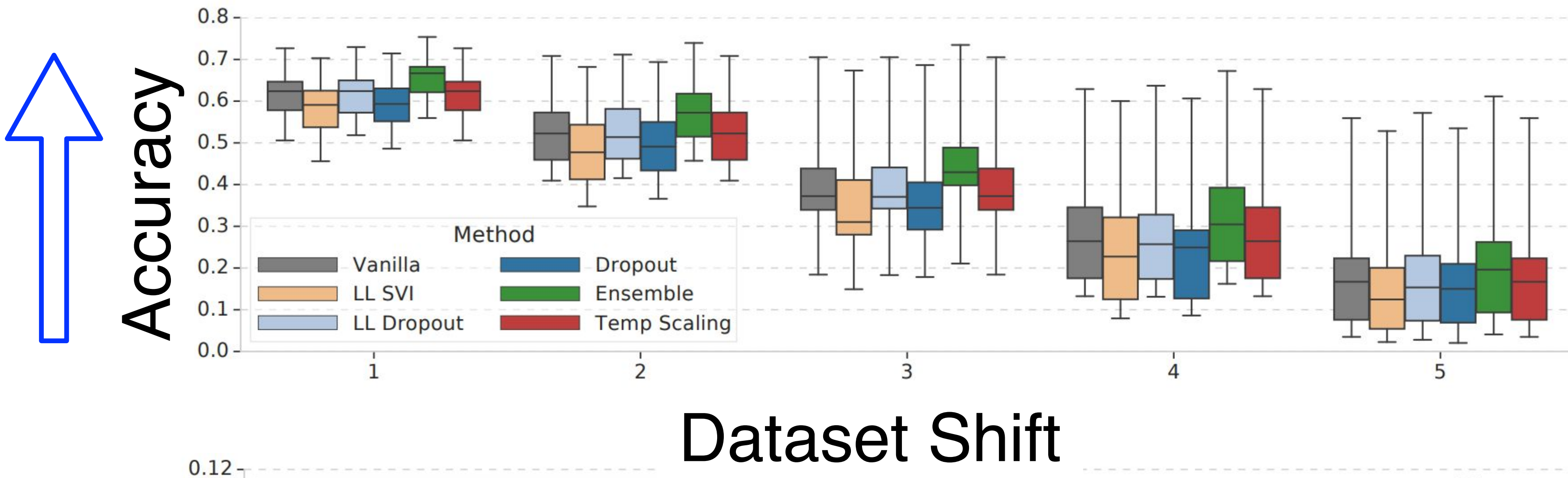
VOGN on ImageNet

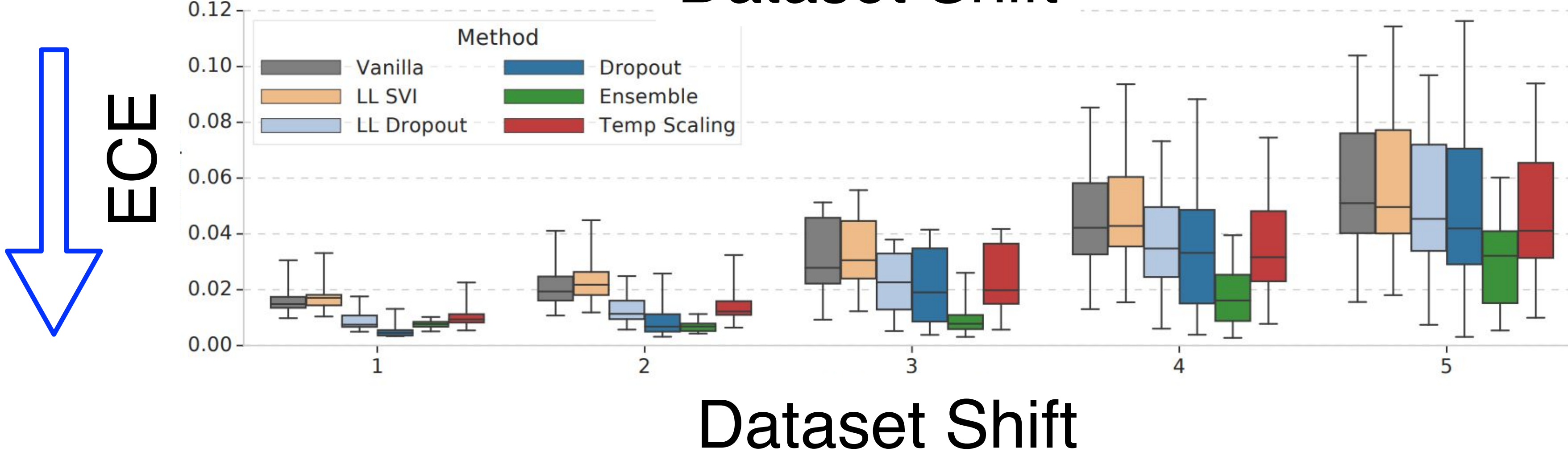
State-of-the-art performance and convergence rate, while preserving benefits of Bayesian principles



1. Osawa et al. “Practical Deep Learning with Bayesian Principles.” NeurIPS (2019).

BDL methods do not really know that they are performing badly under dataset shift





1. Ovadia, Yaniv, et al. "Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift." *NeurIPS* (2019).

Resources for Uncertainty in DL

- Yarin Gal's tutorial (http://bdl101.ml/)
- Benchmarks by OATML (http://bdlb.ml/)

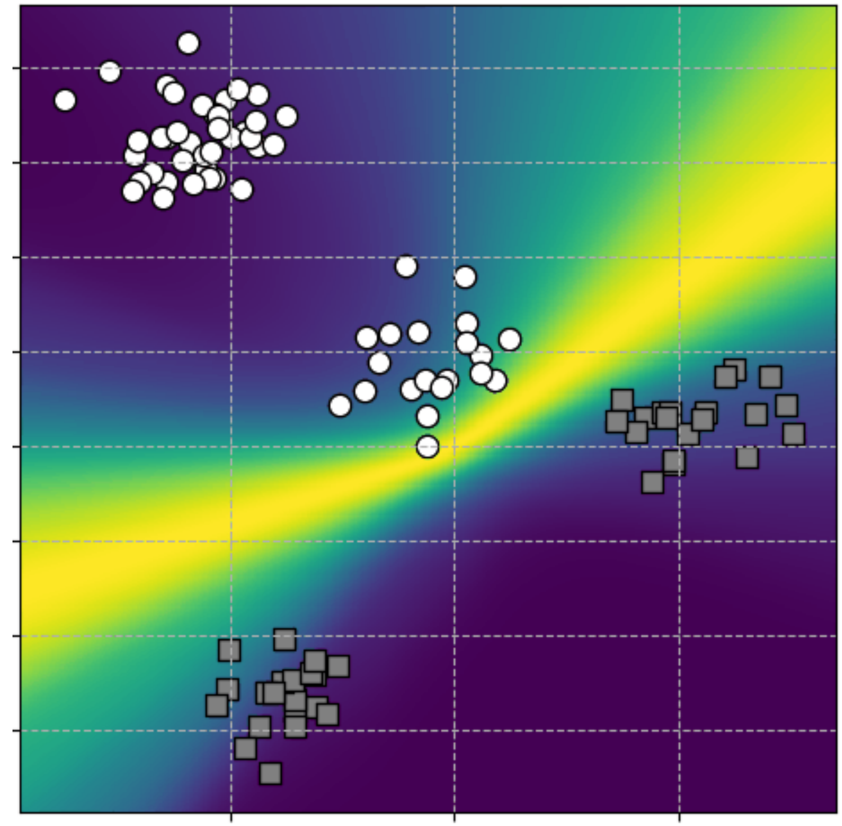
List of Benchmarks

Bayesian Deep Learning Benchmarks (BDL Benchmarks or `bdlb` for short), is an open-source framework that aims to bridge the gap between the design of deep probabilistic machine learning models and their application to real-world problems. Our currently supported benchmarks are:

- [x] Diabetic Retinopathy Diagnosis (in `alpha`, following Leibig et al.)
  - [x] Deterministic
  - [x] Monte Carlo Dropout (following Gal and Ghahramani, 2015)
  - [x] Mean-Field Variational Inference (following Peterson and Anderson, 1987, Wen et al., 2018)
  - [x] Deep Ensembles (following Lakshminarayanan et al., 2016)
  - [x] Ensemble MC Dropout (following Smith and Gal, 2018)
- [ ] Autonomous Vehicle's Scene Segmentation (in `pre-alpha`, following Mukhoti et al.)
- [ ] Galaxy Zoo (in `pre-alpha`, following Walmsley et al.)
- [ ] Fishyscapes (in `pre-alpha`, following Blum et al.)

Challenges in Uncertainty Estimation

- For non convex problem
  - Different local minima correspond to various solutions
  - Local approximations only capture “local uncertainty”
  - Unknown unknowns
- Solutions: More flexible approximations?

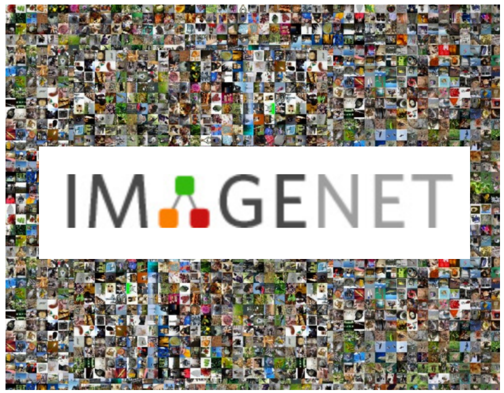


Deep Learning with Bayesian Principles

- Bayesian principles as common principles
  - By computing "posterior approximations"
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and Life-Long learning
- Impact: Many learning-algorithms with a common set of principles.

Continual Life-Long Learning

Standard
Deep
Learning

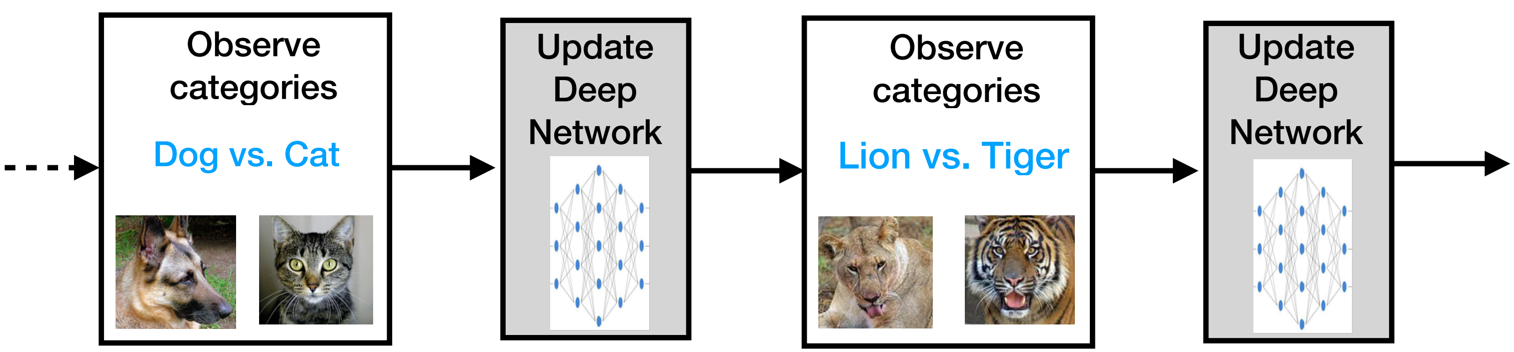


1. Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks." *PNAS* (2017)
2. Parisi et al. "Continual lifelong learning with neural networks: A review." *Neural Networks* (2019)

Continual Life-Long Learning

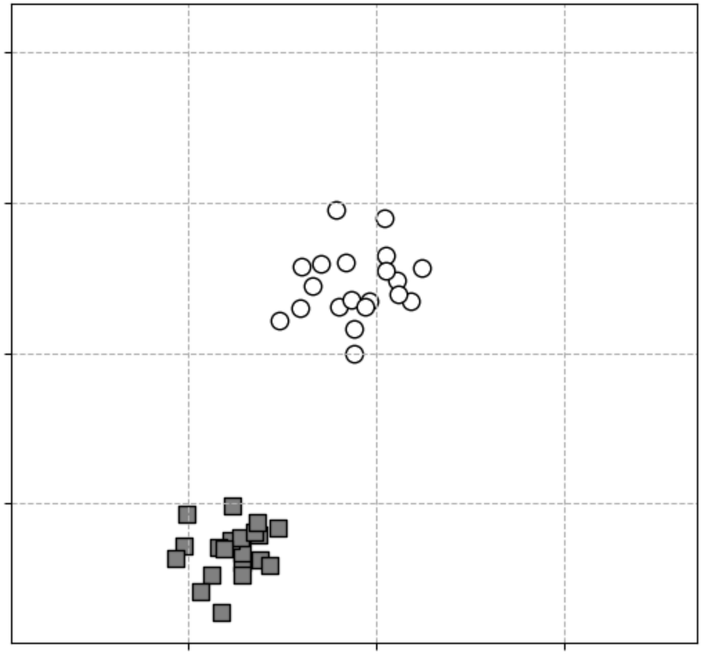
Standard Deep Learning

Continual Learning: past classes never revisited



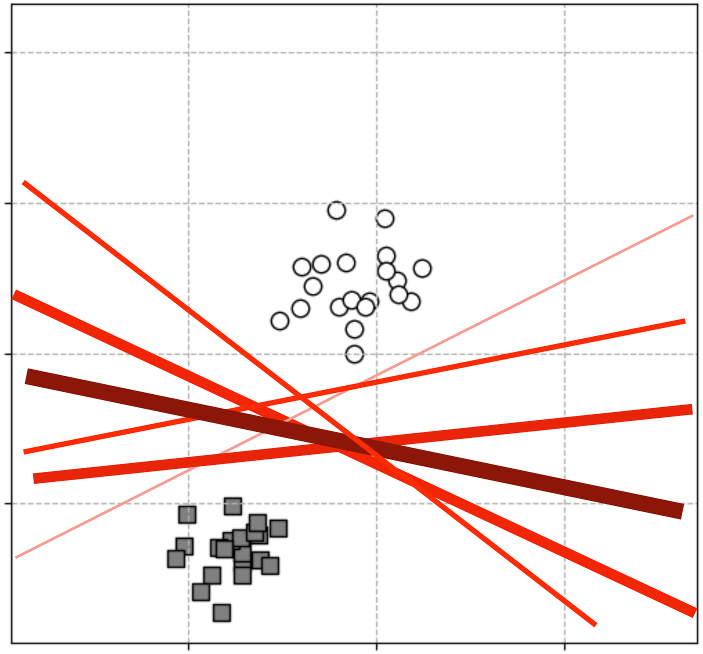
1. Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks." *PNAS* (2017)
2. Parisi et al. "Continual lifelong learning with neural networks: A review." *Neural Networks* (2019)

Life-Long Learning with Bayes



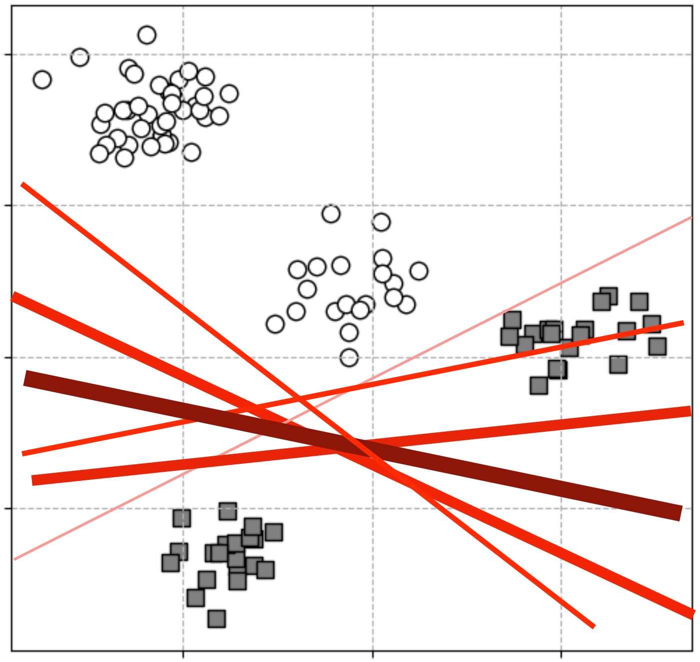
$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)\color{red}{p(\theta)}}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Life-Long Learning with Bayes



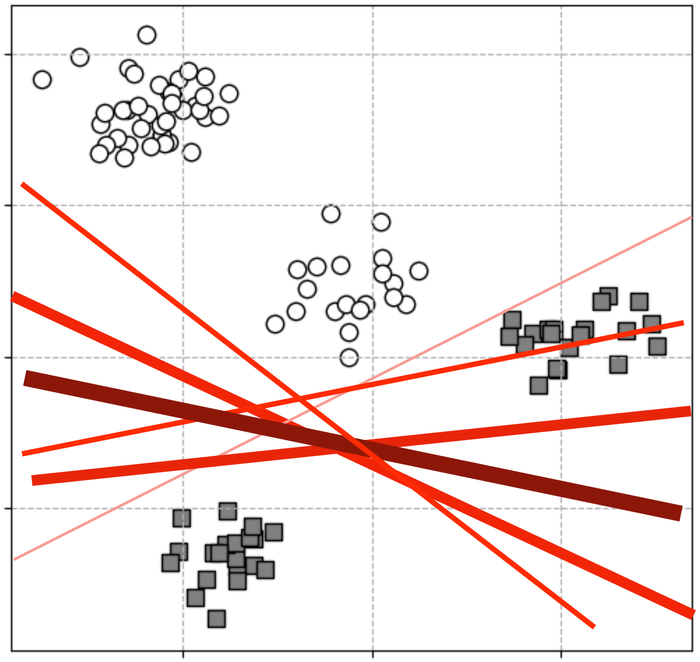
$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)\color{red}{p(\theta)}}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Life-Long Learning with Bayes



$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Life-Long Learning with Bayes

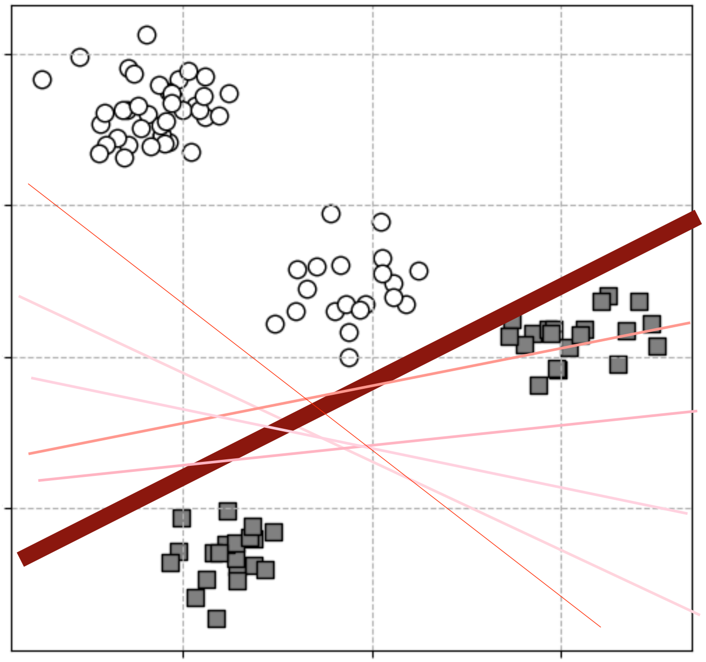


$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Set the prior to the previous posterior and recompute:

$$p(\theta|\mathcal{D}_2, \mathcal{D}_1) = \frac{p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)}{\int p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)d\theta}$$

Life-Long Learning with Bayes

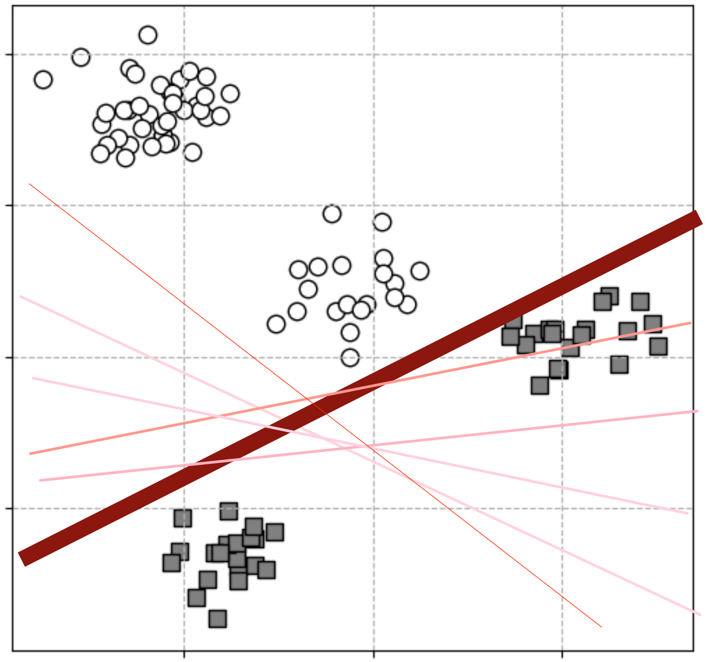


$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Set the prior to the previous posterior and recompute:

$$p(\theta|\mathcal{D}_2, \mathcal{D}_1) = \frac{p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)}{\int p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)d\theta}$$

Life-Long Learning with Bayes



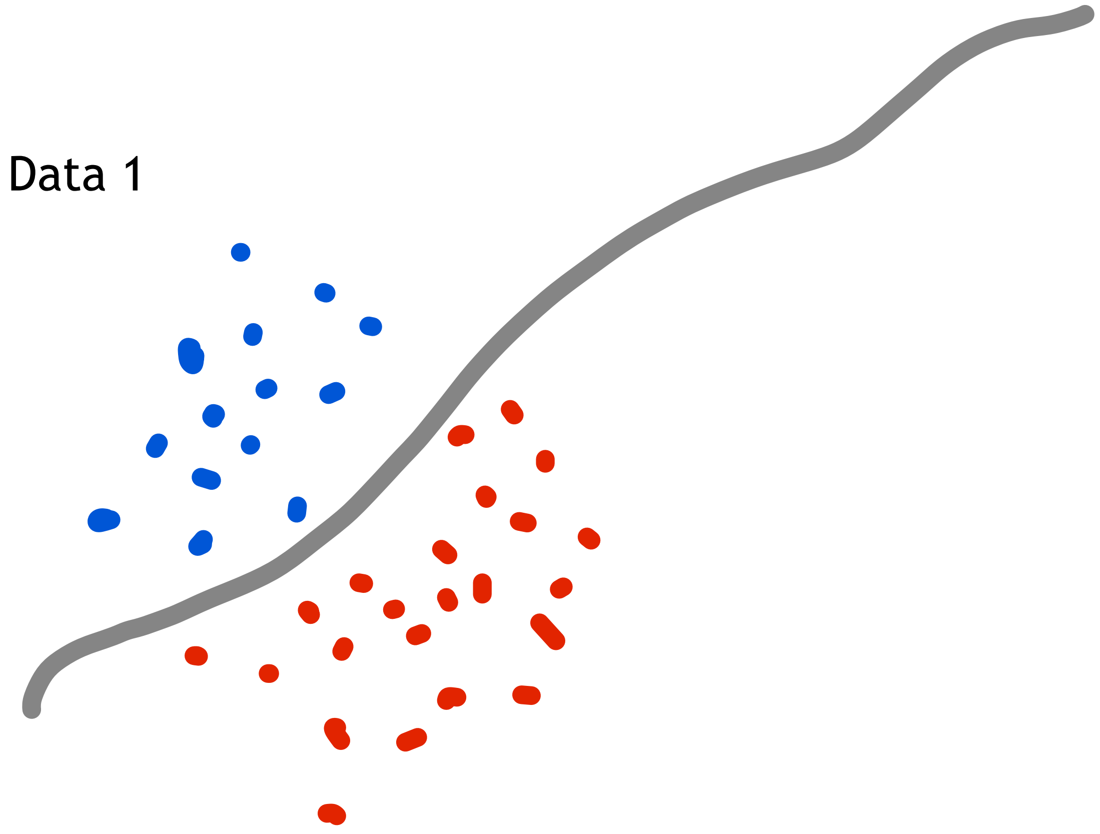
$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Set the prior to the previous posterior and recompute:

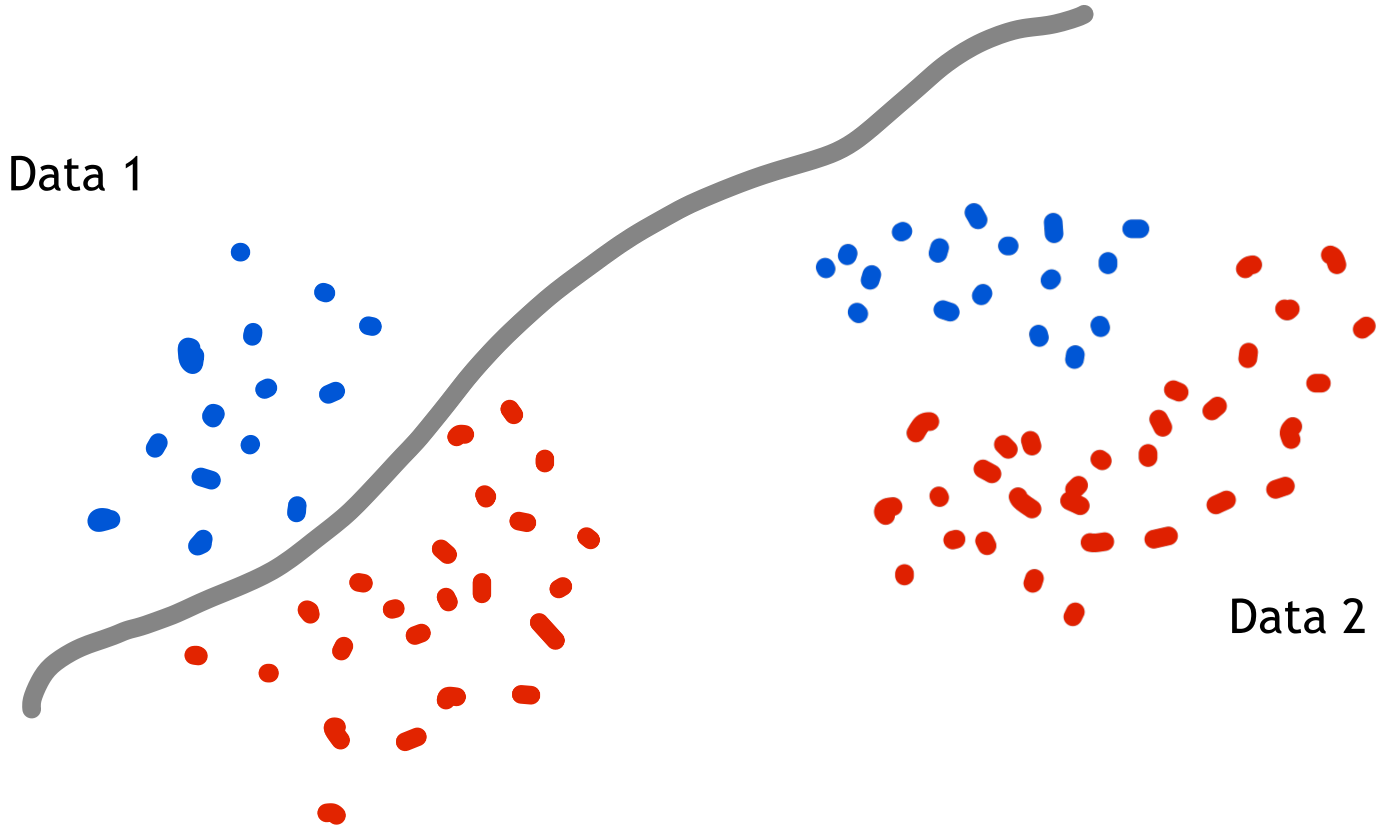
$$p(\theta|\mathcal{D}_2, \mathcal{D}_1) = \frac{p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)}{\int p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)d\theta}$$

Computationally challenging. Approximations do not work well.This is an open problem!

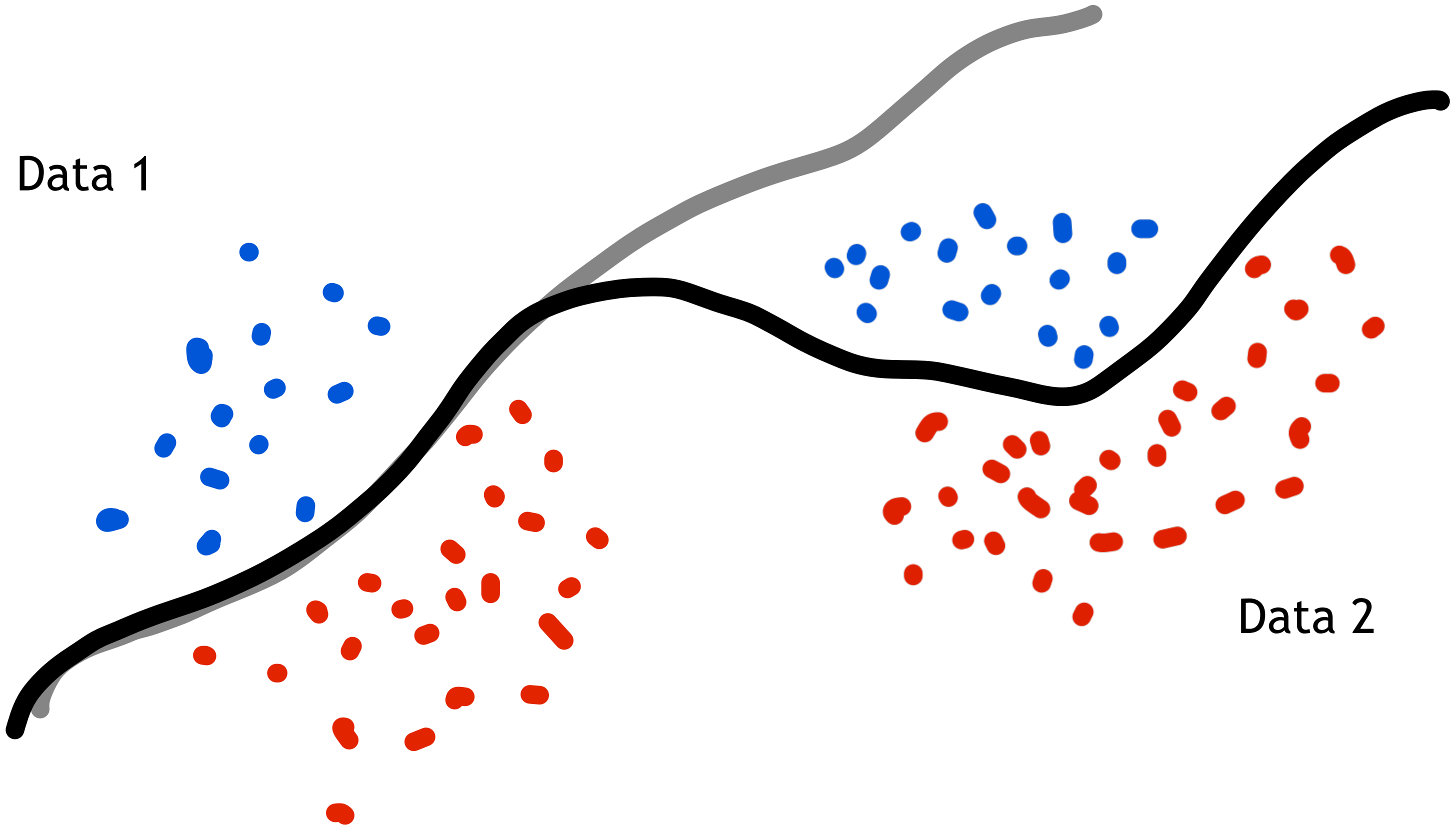
A Key Idea for Life-Long Learning: Posterior Approx in the Function-Space



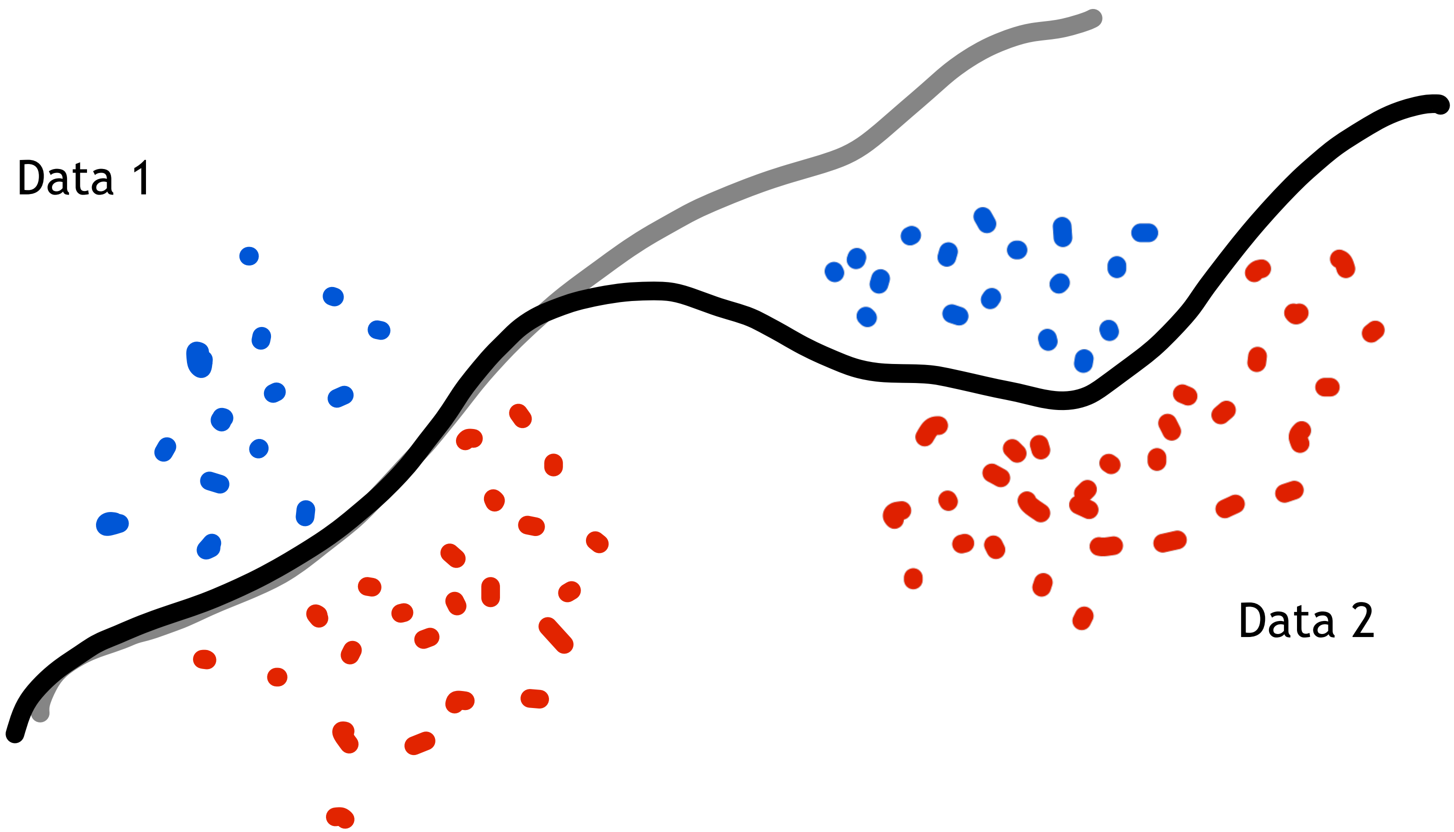
A Key Idea for Life-Long Learning: Posterior Approx in the Function-Space



A Key Idea for Life-Long Learning: Posterior Approx in the Function-Space



A Key Idea for Life-Long Learning: Posterior Approx in the Function-Space



Change the network weights to match the network output (function) at Data 1 while classifying Data 2

Life-Long Learning with Bayesian Principles

- Connect the weight and function spaces.
  - Cheap algorithms to train in the weight space while regularizing in the function space.
- Background
  - Linear models and Gaussian Process (GP)
  - Neural Nets and GPs (requires infinite-width nets)
- DNN2GP
  - Convert trained finite-widths nets to GPs
  - Convert the iterates of DL algorithms to GPs
- Applications to Continual Learning

Linear model and GPs

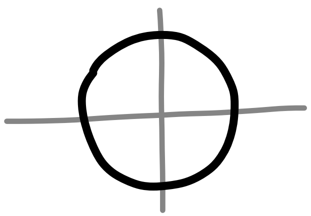
Gaussian prior on weights induces GP prior on functions

Chapter 2 in Rasmussen and Williams, Gaussian processes for Machine Learning, 2006

Linear model and GPs

Gaussian prior on weights induces GP prior on functions

$w \sim \mathcal{N}(0, I)$



feature

function

$$f(x) = \phi(x)^T w$$

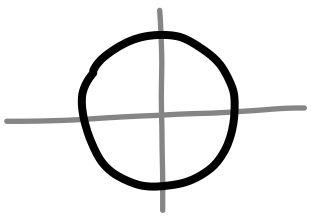
weights

Chapter 2 in Rasmussen and Williams, Gaussian processes for Machine Learning, 2006

Linear model and GPs

Gaussian prior on weights induces GP prior on functions

$$w \sim \mathcal{N}(0, I)$$



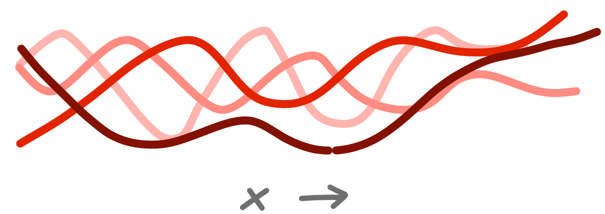
function

feature

$$f(x) = \phi(x)^T w$$

weights

$$f(x) \sim GP\big(\underbrace{0}_{\text{mean}}, \underbrace{\phi(x)^T \phi(x')}_{\text{kernel}}\big)$$

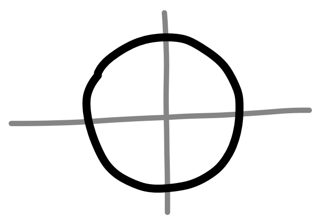


Chapter 2 in Rasmussen and Williams, Gaussian processes for Machine Learning, 2006

Linear model and GPs

Gaussian prior on weights induces GP prior on functions

$w \sim \mathcal{N}(0, I)$



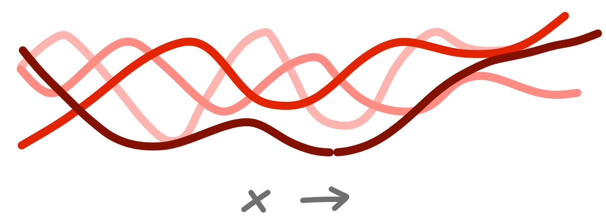
function

feature

$f(x) = \phi(x)^T w$

weights

$f(x) \sim GP\Big(\underbrace{0}_{\text{mean}}, \underbrace{\phi(x)^T \phi(x')}_{\text{kernel}}\Big)$



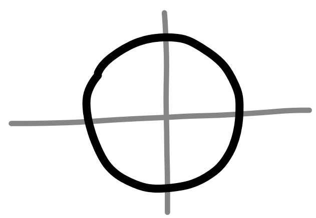
Gaussian posterior on w induces a GP posterior on f

Chapter 2 in Rasmussen and Williams, Gaussian processes for Machine Learning, 2006

Linear model and GPs

Gaussian prior on weights induces GP prior on functions

$w \sim \mathcal{N}(0, I)$



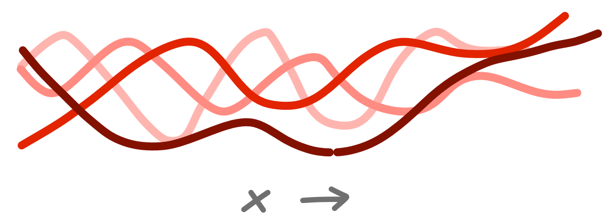
function

feature

$f(x) = \phi(x)^T w$

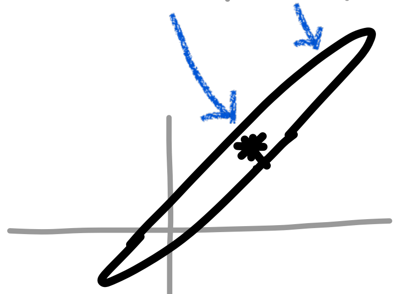
weights

$f(x) \sim GP\Big(\underbrace{0}_{\text{mean}}, \underbrace{\phi(x)^T \phi(x')}_{\text{kernel}}\Big)$



Gaussian posterior on w induces a GP posterior on f

$w \sim \mathcal{N}(w_*, \Sigma_*)$

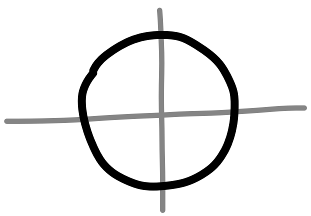


Chapter 2 in Rasmussen and Williams, Gaussian processes for Machine Learning, 2006

Linear model and GPs

Gaussian prior on weights induces GP prior on functions

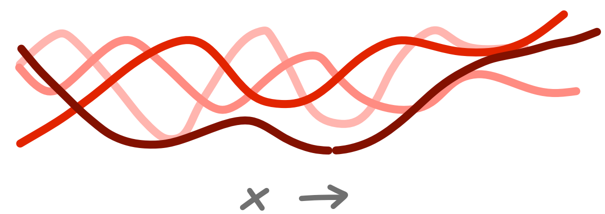
$w \sim \mathcal{N}(0, I)$



function, feature

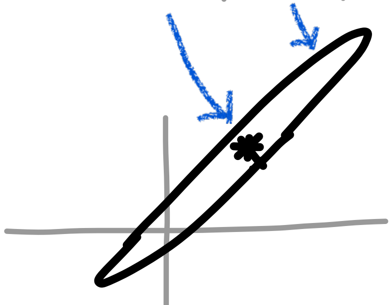
$$\underbrace{f(x)}_{\text{function}} = \overbrace{\phi(x)}^{\text{feature}}{}^T \underbrace{w}_{\text{weights}}$$

$$f(x) \sim GP(\underbrace{0}_{\text{mean}}, \underbrace{\phi(x)^T \phi(x')}_{\text{kernel}})$$

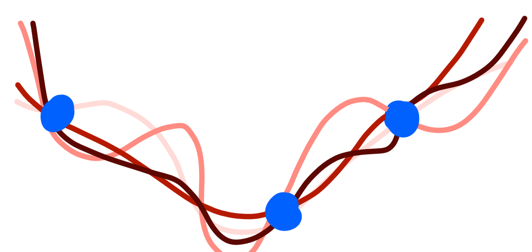


Gaussian posterior on w induces a GP posterior on f

$w \sim \mathcal{N}(w_*, \Sigma_*)$



$$f(x) \sim GP(\phi(x)^T w_*, \phi(x)^T \Sigma_* \phi(x'))$$



Chapter 2 in Rasmussen and Williams, Gaussian processes for Machine Learning, 2006

Deep Networks and GPs

Gaussian prior on weights induces GP prior on functions

Lee et al., Deep Neural Networks as Gaussian Processes, ICLR 2018 (and many more…)

Deep Networks and GPs

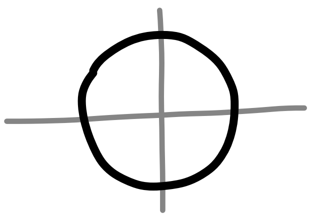
Gaussian prior on weights induces GP prior on functions

$w \sim \mathcal{N}(0, I)$

DNN

$f_w(x)$

weights



Lee et al., Deep Neural Networks as Gaussian Processes, ICLR 2018 (and many more…)

Deep Networks and GPs

Gaussian prior on weights induces GP prior on functions

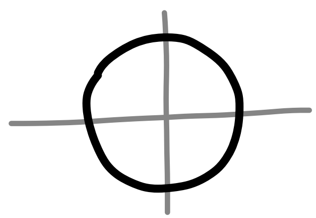
$w \sim \mathcal{N}(0, I)$

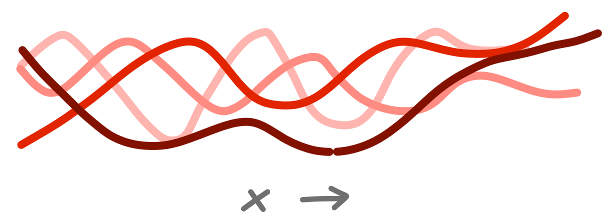
DNN

$f_w(x)$

weights

$f_w(x) \xrightarrow[\text{as width} \to \infty]{\text{converges to}} f(x) \sim GP\big(\underbrace{0}_{\text{mean}}, \underbrace{k(x,x')}_{\text{kernel}}\big)$



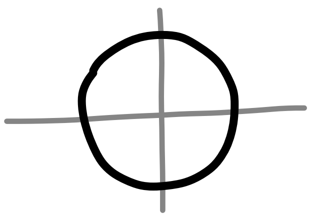


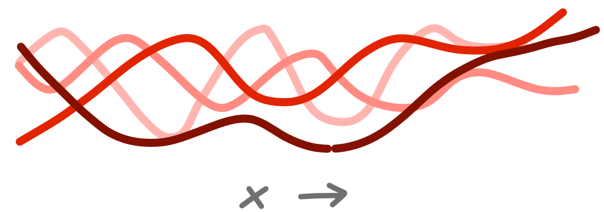
Lee et al., Deep Neural Networks as Gaussian Processes, ICLR 2018 (and many more…)

Deep Networks and GPs

Gaussian prior on weights induces GP prior on functions

$$w \sim \mathcal{N}(0, I) \qquad \underset{\text{weights}}{\overset{\text{DNN}}{f_w(x)}} \xrightarrow[\text{as width} \to \infty]{\text{converges to}} f(x) \sim GP\Big(\underbrace{0}_{\text{mean}}, \underbrace{k(x,x')}_{\text{kernel}}\Big)$$





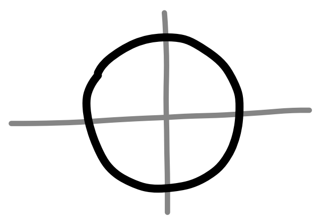
Q: Does this hold at finite width? And for posteriors?

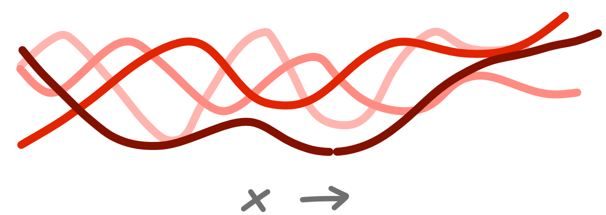
Lee et al., Deep Neural Networks as Gaussian Processes, ICLR 2018 (and many more…)

Deep Networks and GPs

Gaussian prior on weights induces GP prior on functions

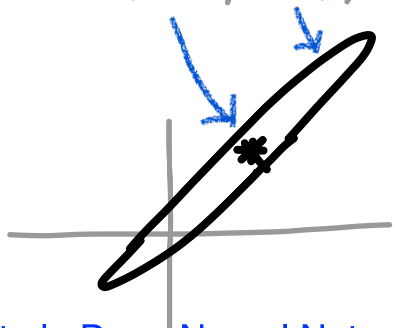
$w \sim \mathcal{N}(0, I)$ DNN $\underbrace{f_w}_{\text{weights}}(x) \xrightarrow[\text{as width} \to \infty]{\text{Converges to}} f(x) \sim GP(\underbrace{0}_{\text{mean}}, \underbrace{k(x,x')}_{\text{kernel}})$

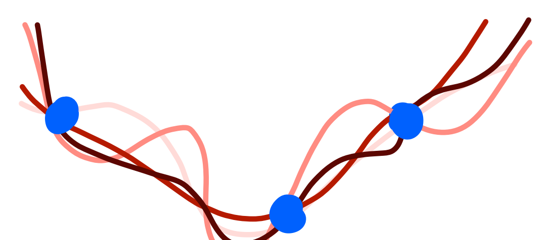




Q: Does this hold at finite width? And for posteriors?

$w \sim \mathcal{N}(w_*, \Sigma_*)$ $\xrightarrow{?}$ $f(x) \sim GP(\phi(x)^T w_*, \phi(x)^T \Sigma_* \phi(x'))$





Lee et al., Deep Neural Networks as Gaussian Processes, ICLR 2018 (and many more…)

DNN2GP for regression

Using DNN2GP, we can convert a trained network into GP

$w \sim \mathcal{N}(w_*, \Sigma_*)$

$f(x) \sim GP\left(f_{w_*}(x),\ J_{w_*}(x)\,\Sigma_*\,J_{w_*}(x')^\top\right)$

Khan et al., Approximate Inference Turns Deep Networks into Gaussian Processes, NeurIPS, 2019

DNN2GP for regression

Using DNN2GP, we can convert a trained network into GP

$$w_* = \arg\min_w \underbrace{\sum_{i=1}^{N} \left(y_i - f_w(x_i)\right)^2}_{\text{squared loss}} + \underbrace{\delta\, w^T w}_{L_2 \text{ prior}}$$

$$w \sim \mathcal{N}(w_*, \Sigma_*)$$

$$f(x) \sim GP\left(f_{w_*}(x),\ J_{w_*}(x)\, \Sigma_*\, J_{w_*}(x')\right)$$

Khan et al., Approximate Inference Turns Deep Networks into Gaussian Processes, NeurIPS, 2019

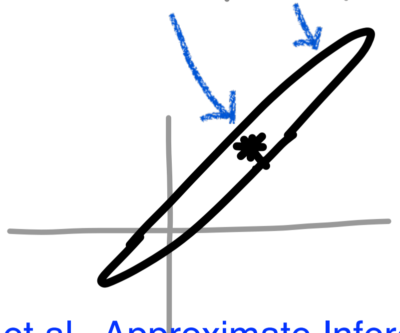
DNN2GP for regression

Using DNN2GP, we can convert a trained network into GP

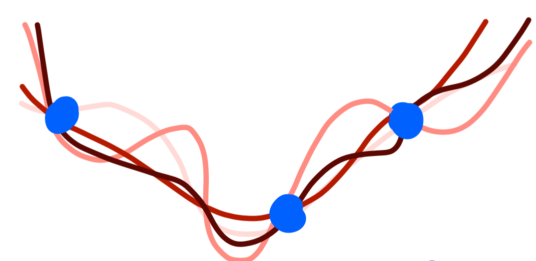
$$w_* = \arg\min_w \sum_{i=1}^{N} \underbrace{\left(y_i - f_w(x_i)\right)^2}_{\text{squared loss}} + \underbrace{\delta\, w^T w}_{L_2 \text{ prior}}$$

$$\Sigma_*^{-1} = \sum_{i=1}^{N} \nabla_w f_{w_*}(x_i) \underbrace{\nabla_w f_{w_*}(x_i)}_{J_{w_*}(x_i) \,\leftarrow\, \text{Jacobian}}{}^T + \delta I \quad \leftarrow \text{Gauss-Newto Curvature}$$

$$w \sim \mathcal{N}(w_*, \Sigma_*)$$



$$f(x) \sim GP\left(f_{w_*}(x),\ J_{w_*}(x)\, \Sigma_*\, J_{w_*}(x')\right)$$



Khan et al., Approximate Inference Turns Deep Networks into Gaussian Processes, NeurIPS, 2019

DNN2GP Generalization

This generalizes to twice differentiable loss and priors

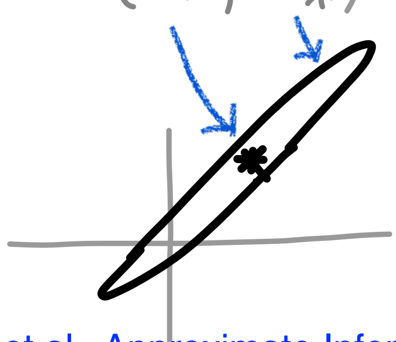
Link function

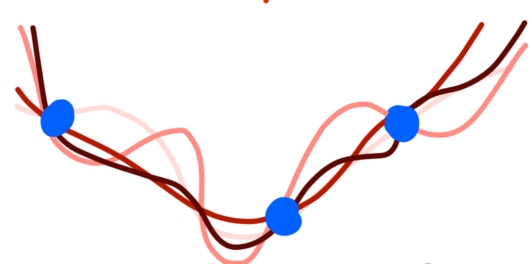
$$w_* = \arg\min_w \sum_{i=1}^N \underbrace{\ell(y_i, \sigma(f_w(x_i)))}_{\text{Diff. loss}} + \underbrace{\delta R(w)}_{\text{Convex prior}}$$

$$\Sigma_*^{-1} = \sum_{i=1}^N \nabla_w f_{w_*}(x_i) \underbrace{\nabla^2_{ff} \ell(\ )}_{\Lambda_{w_*}(x)} \nabla_w f_{w_*}(x_i)^T + \delta \nabla^2_{ww} R(w) \leftarrow \text{Generalized Gauss-Newton Curvature}$$

$$w \sim \mathcal{N}(w_*, \Sigma_*)$$

$$f(x) \sim GP\left(\sigma(f_{w_*}(x)), \lambda_{w_*}(x) J_{w_*}(x) \Sigma_* J_{w_*}(x') \lambda_{w_*}(x)\right)$$





Khan et al., Approximate Inference Turns Deep Networks into Gaussian Processes, NeurIPS, 2019

Deep Learning as GP inference

Iterations of algorithms too can be written as GP inference

$$w_{t+1} \leftarrow w_t - \rho \, \underbrace{(S_t + \delta I)^{-1}}_{\Sigma_t} \left[\sum_{i \in \mathcal{M}} g_i \right] \leftarrow \text{minibatch gradient}$$

$$\text{Scale matrix} \rightarrow S_{t+1} \leftarrow (1-\beta) \, S_t + \sum_{i \in \mathcal{M}} J_{w_t}(x_i) \, J_{w_t}(x_i)^T \leftarrow \text{minibatch GN Curvature}$$

Khan et al., Approximate Inference Turns Deep Networks into Gaussian Processes, NeurIPS, 2019

Deep Learning as GP inference

Iterations of algorithms too can be written as GP inference

$$w_{t+1} \leftarrow w_t - \rho \underbrace{(S_t + \delta I)^{-1}}_{\Sigma_t} \left[\sum_{i \in \mathcal{M}} g_i \right] \leftarrow \text{minibatch gradient}$$

$$\text{Scale matrix} \rightarrow S_{t+1} \leftarrow (1-\beta) S_t + \sum_{i \in \mathcal{M}} J_{w_t}(x_i) J_{w_t}(x_i)^T \leftarrow \text{minibatch GN Curvature}$$

Training in w space induces a sequence in f space

Khan et al., Approximate Inference Turns Deep Networks into Gaussian Processes, NeurIPS, 2019

Deep Learning as GP inference

Iterations of algorithms too can be written as GP inference

$$w_{t+1} \leftarrow w_t - \rho \underbrace{(S_t+\delta I)^{-1}}_{\Sigma_t} \left[\sum_{i\in M} g_i\right]$$ ← minibatch gradient

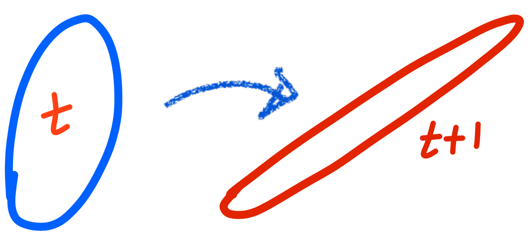
Scale matrix → $$s_{t+1} \leftarrow (1-\rho)\, S_t + \sum_{i\in M} J_{w_t}(x_i)\, J_{w_t}(x_i)^T$$ ← minibatch GN Curvature

Training in w space induces a sequence in f space

$$\mathcal{N}(w_t, \Sigma_t)$$

$$\downarrow$$

$$\mathcal{N}(w_{t+1}, \Sigma_{t+1})$$



Khan et al., Approximate Inference Turns Deep Networks into Gaussian Processes, NeurIPS, 2019

Deep Learning as GP inference

Iterations of algorithms too can be written as GP inference

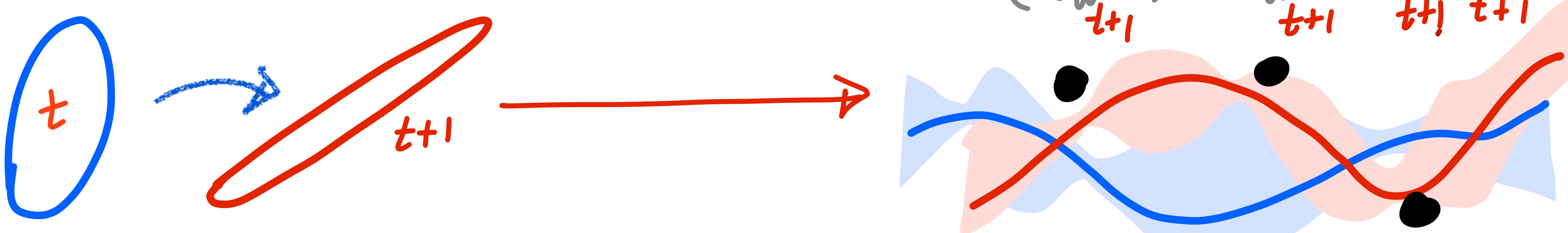
$$w_{t+1} \leftarrow w_t - \rho \underbrace{(S_t+\delta I)^{-1}}_{\Sigma_t} \Big[\sum_{i\in\mathcal{M}} g_i\Big] \leftarrow \text{minibatch gradient}$$

$$\text{Scale matrix} \rightarrow S_{t+1} \leftarrow (1-\rho)\, S_t + \sum_{i\in\mathcal{M}} J_{w_t}(x_i)\, J_{w_t}(x_i)^T \leftarrow \text{minibatch GN Curvature}$$

Training in w space induces a sequence in f space

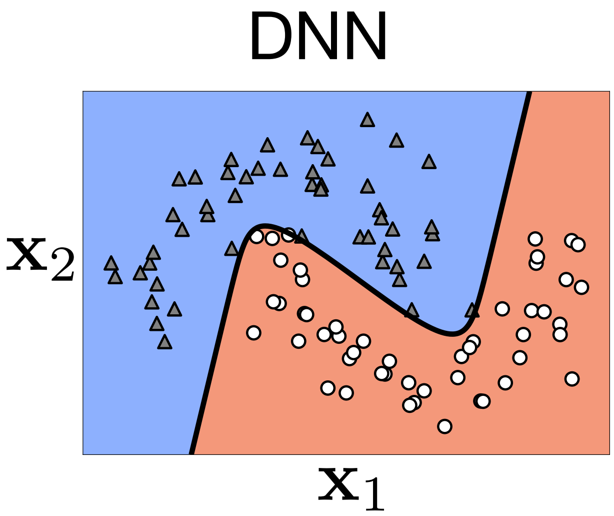
$$\mathcal{N}(w_t, \Sigma_t) \qquad \mathcal{GP}\big(f_{w_t}(x) \quad J_{w_t}(x)\,\Sigma_t\,J_{w_t}(x')\big)$$

$$\mathcal{N}(w_{t+1}, \Sigma_{t+1}) \qquad \mathcal{GP}\big(f_{w_{t+1}}(x) \quad J_{w_{t+1}}(x)\,\Sigma_{t+1}\,J_{w_{t+1}}(x')\big)$$



Khan et al., Approximate Inference Turns Deep Networks into Gaussian Processes, NeurIPS, 2019

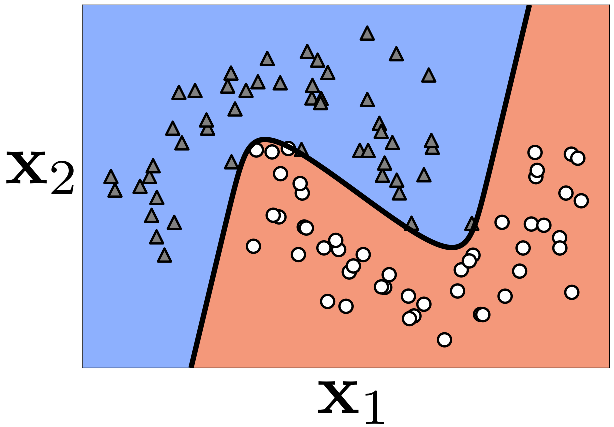
Outline of the derivation



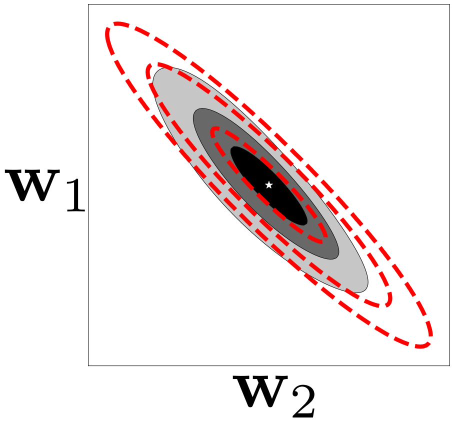
Khan et al., Approximate Inference Turns Deep Networks into Gaussian Processes, NeurIPS, 2019

Outline of the derivation

DNN



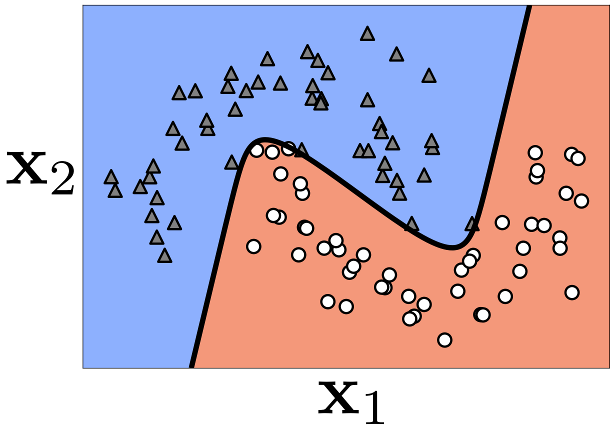
Posterior Approx.



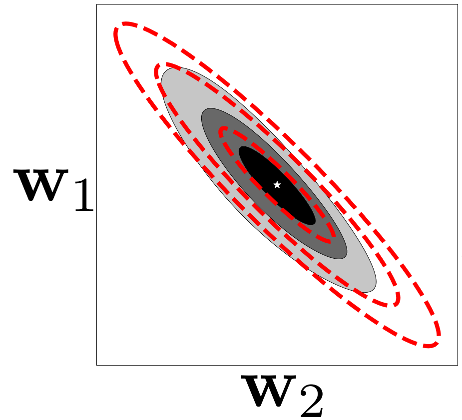
Khan et al., Approximate Inference Turns Deep Networks into Gaussian Processes, NeurIPS, 2019

Outline of the derivation

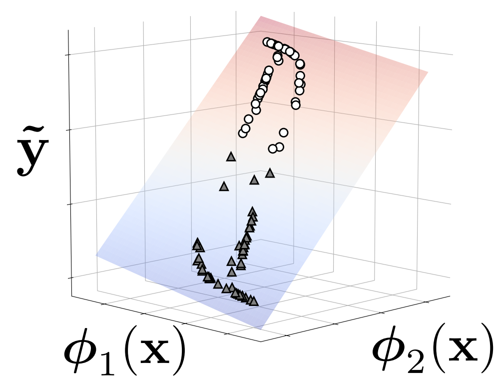
DNN



Posterior Approx.

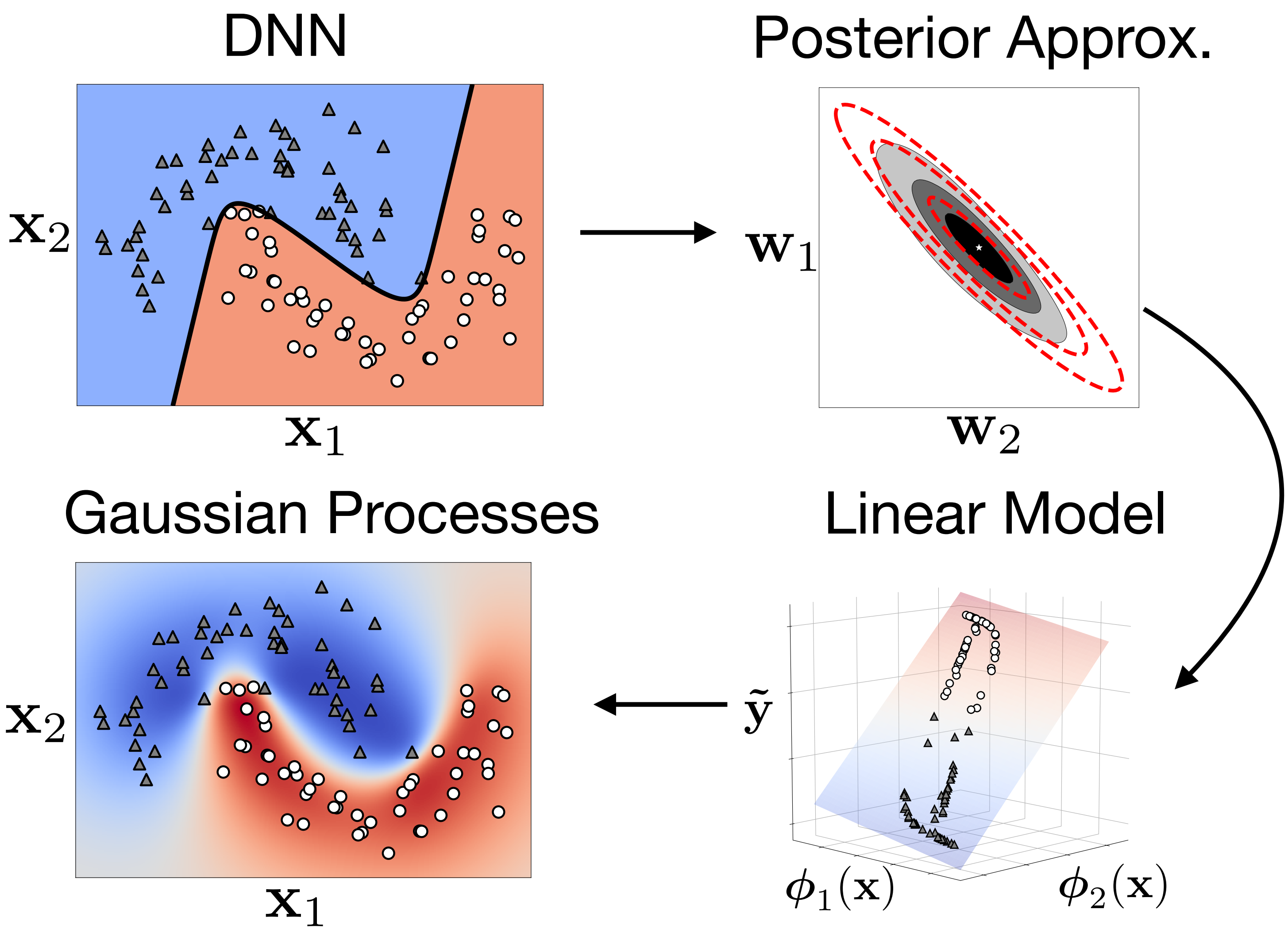


Linear Model



Khan et al., Approximate Inference Turns Deep Networks into Gaussian Processes, NeurIPS, 2019

Outline of the derivation



Khan et al., Approximate Inference Turns Deep Networks into Gaussian Processes, NeurIPS, 2019

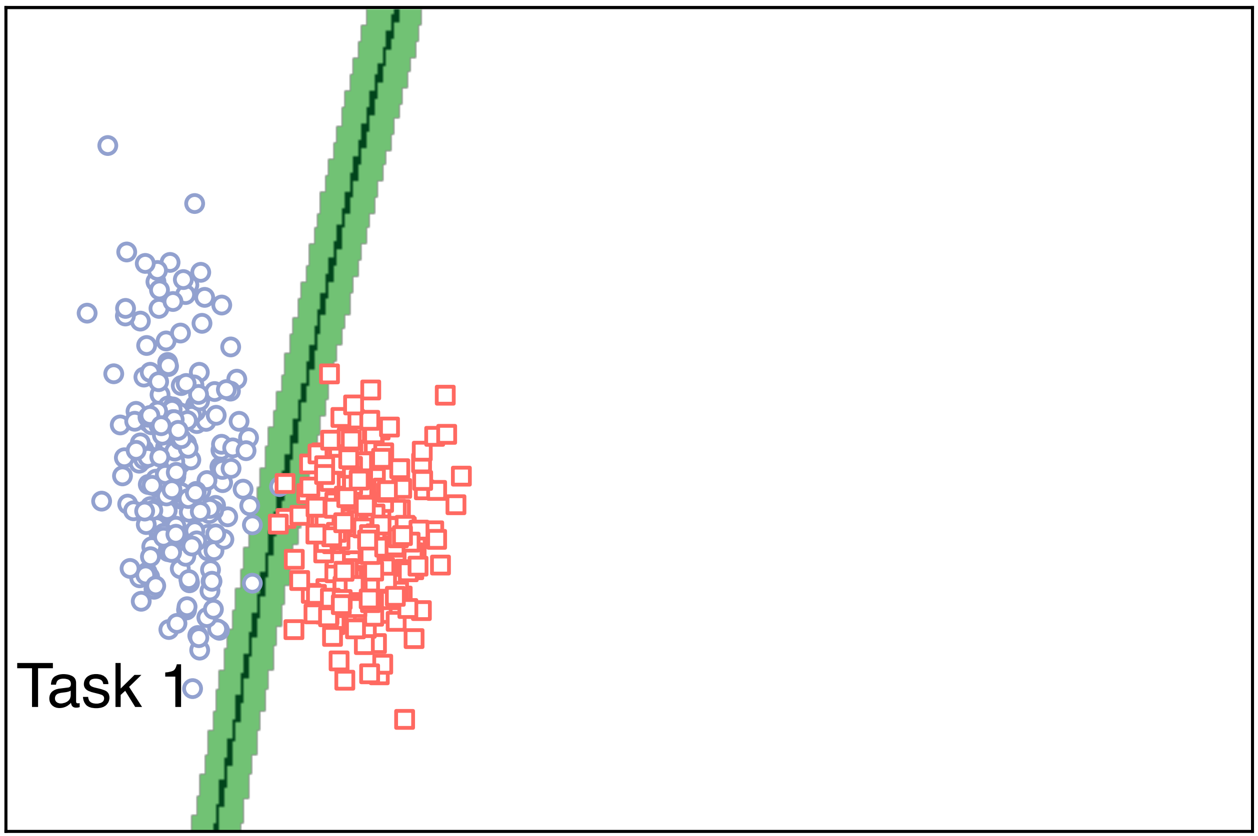
Functional Regularization of Memorable Past (FROMP)

Identify, memorize, and regularize the past obtained using DNN2GP

Pan et al. Continual Deep Learning by Functional Regularisation of Memorable Past, ArXiv 2020

Functional Regularization of Memorable Past (FROMP)

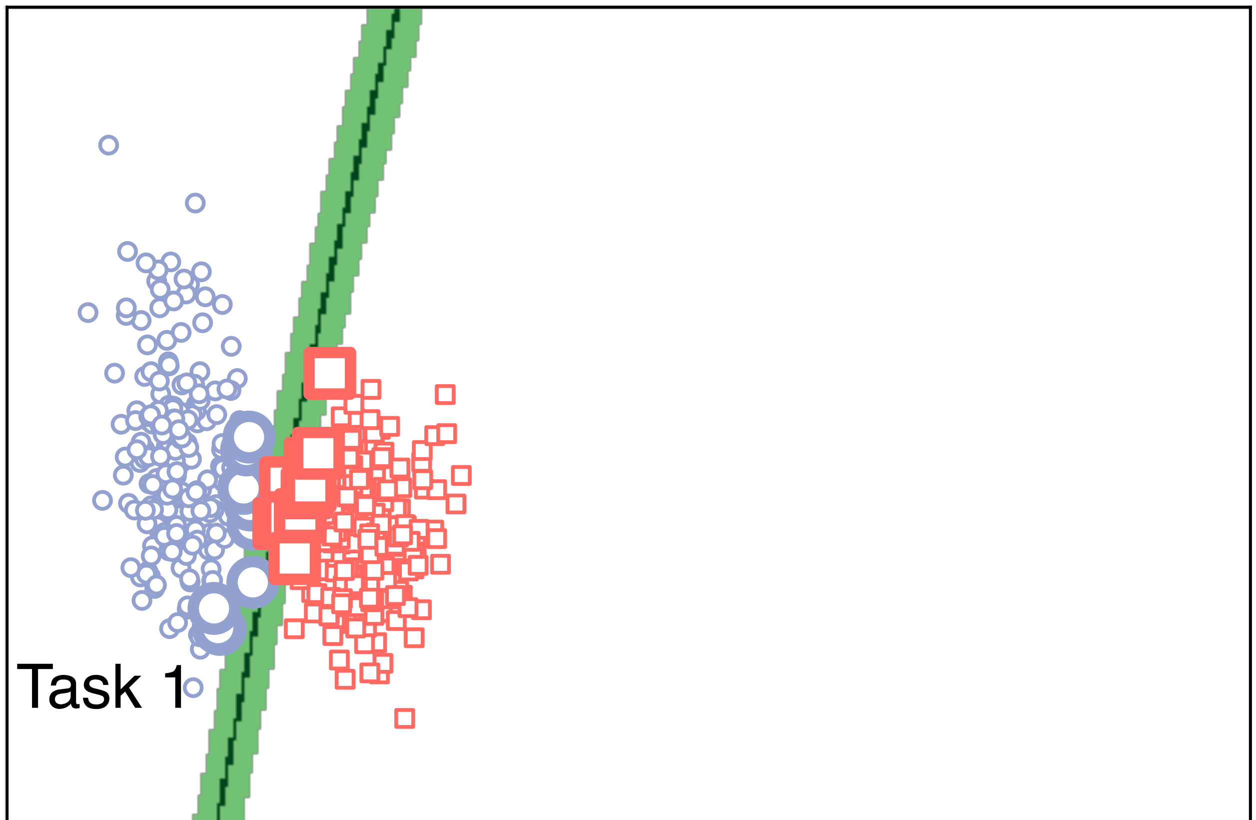
Identify, memorize, and regularize the past obtained using DNN2GP



Pan et al. Continual Deep Learning by Functional Regularisation of Memorable Past, ArXiv 2020

Functional Regularization of Memorable Past (FROMP)

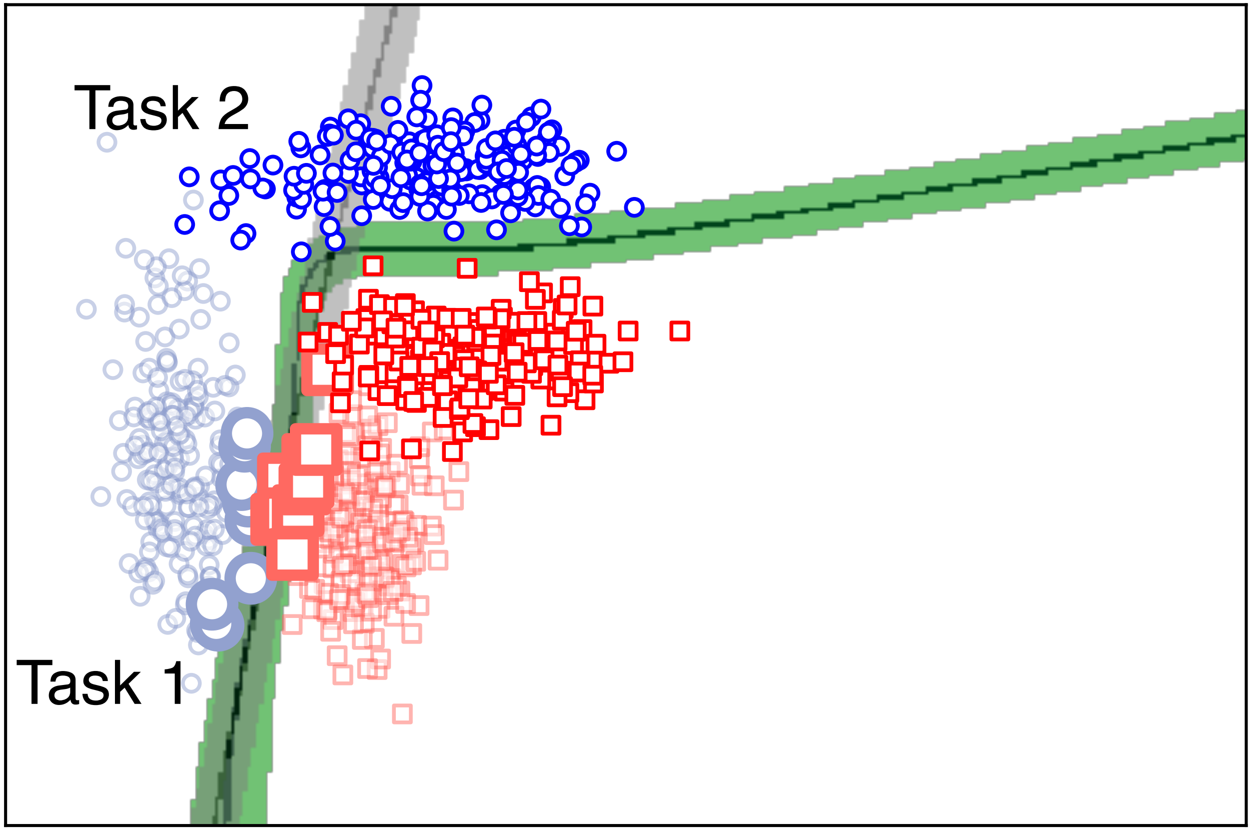
Identify, memorize, and regularize the past obtained using DNN2GP



Pan et al. Continual Deep Learning by Functional Regularisation of Memorable Past, ArXiv 2020

Functional Regularization of Memorable Past (FROMP)

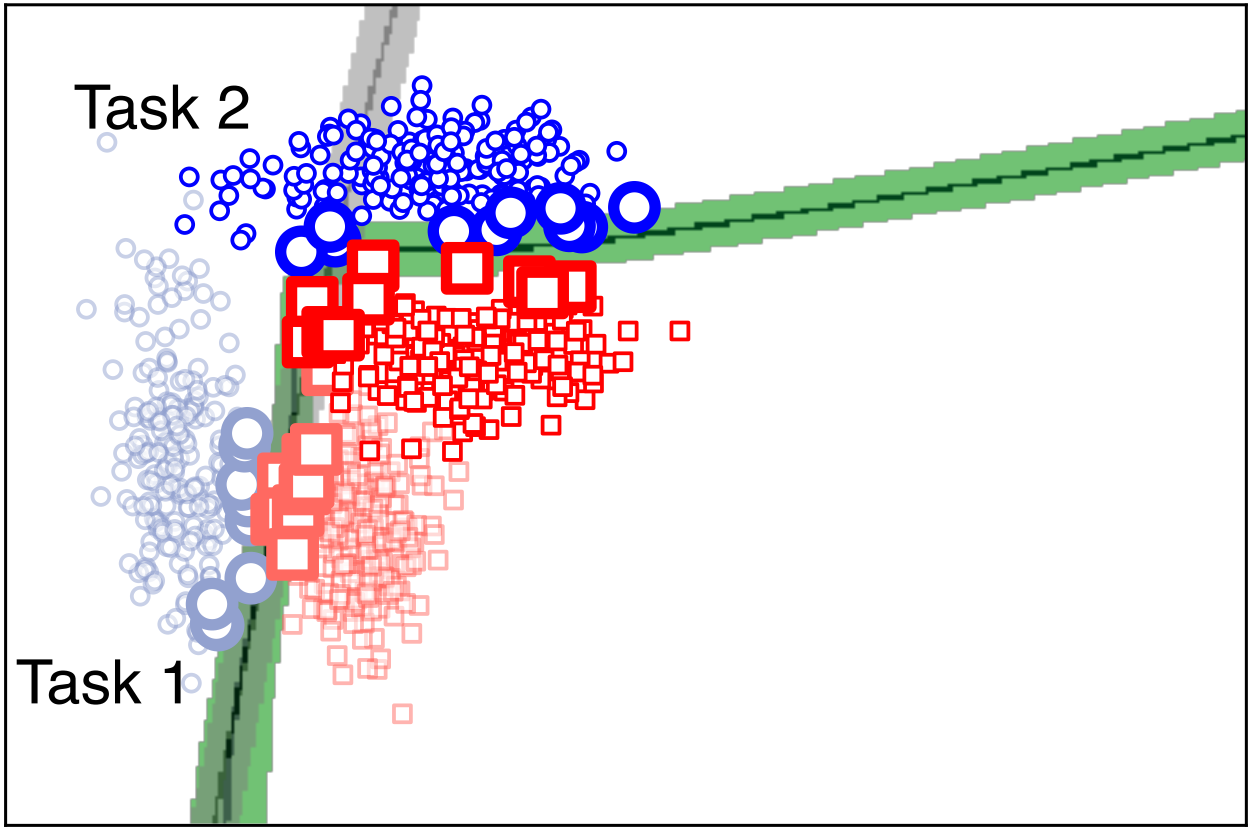
Identify, memorize, and regularize the past obtained using DNN2GP



Pan et al. Continual Deep Learning by Functional Regularisation of Memorable Past, ArXiv 2020

Functional Regularization of Memorable Past (FROMP)

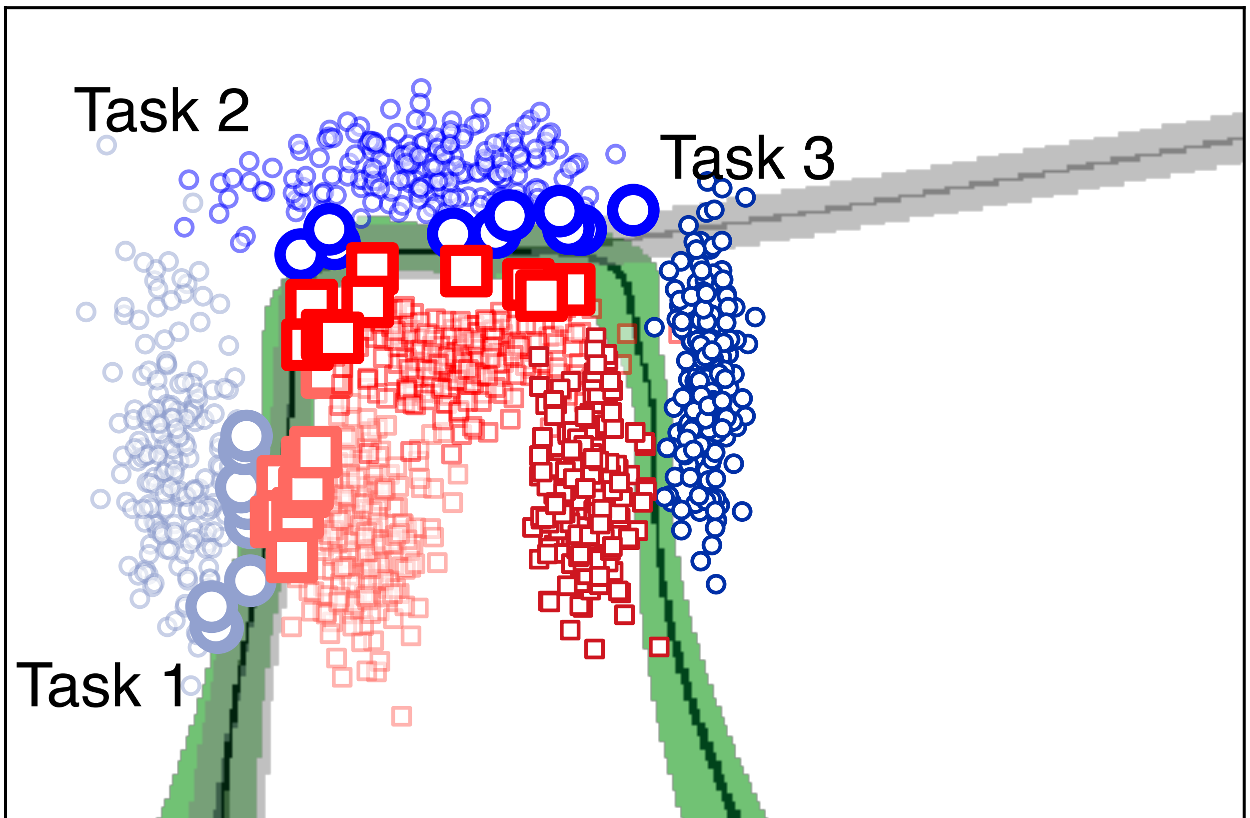
Identify, memorize, and regularize the past obtained using DNN2GP



Pan et al. Continual Deep Learning by Functional Regularisation of Memorable Past, ArXiv 2020

Functional Regularization of Memorable Past (FROMP)

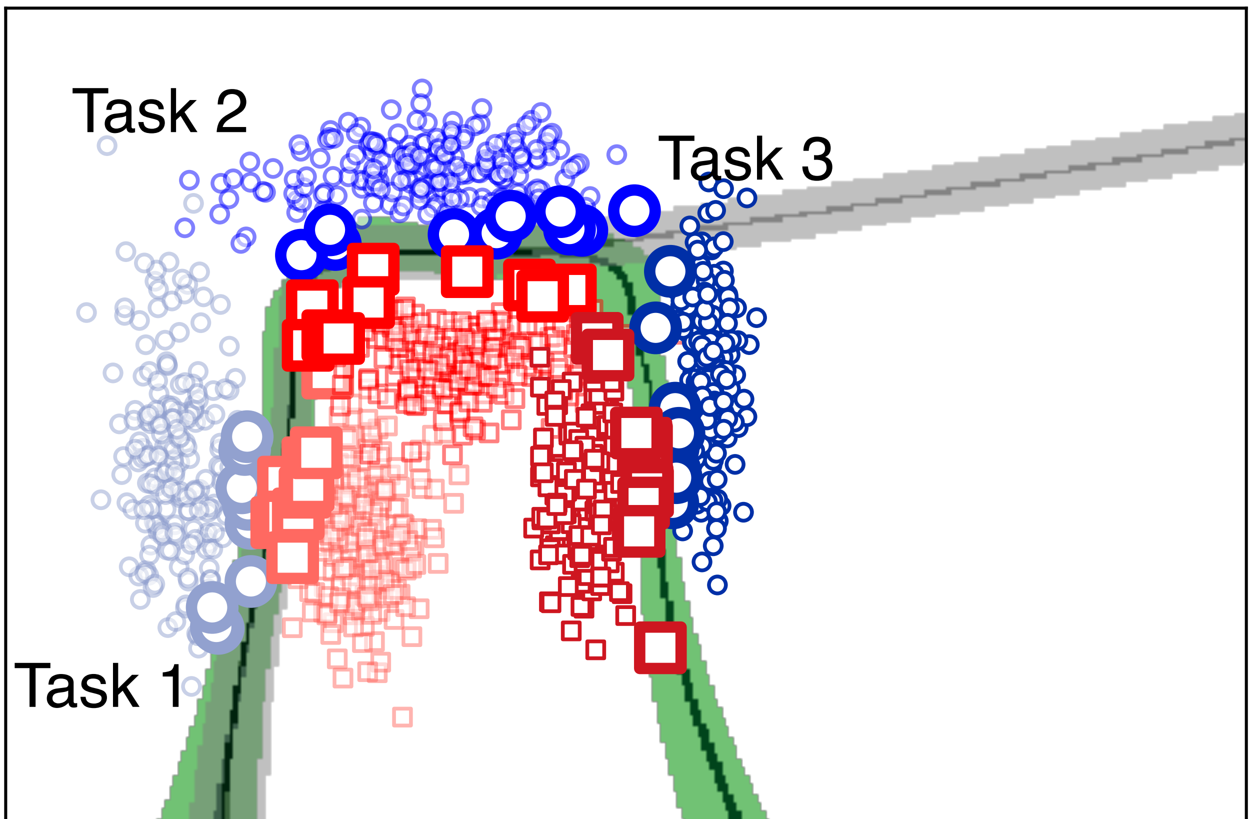
Identify, memorize, and regularize the past obtained using DNN2GP



Pan et al. Continual Deep Learning by Functional Regularisation of Memorable Past, ArXiv 2020

Functional Regularization of Memorable Past (FROMP)

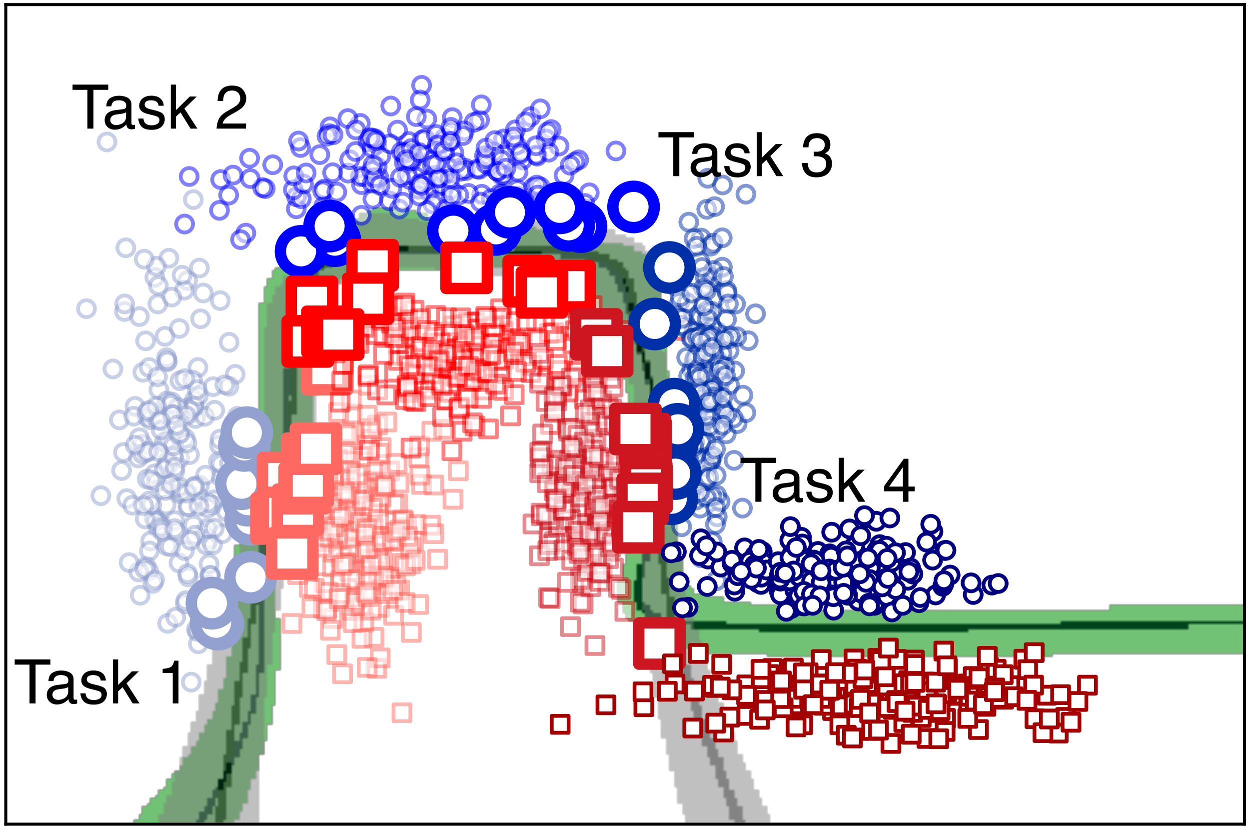
Identify, memorize, and regularize the past obtained using DNN2GP



Pan et al. Continual Deep Learning by Functional Regularisation of Memorable Past, ArXiv 2020

Functional Regularization of Memorable Past (FROMP)

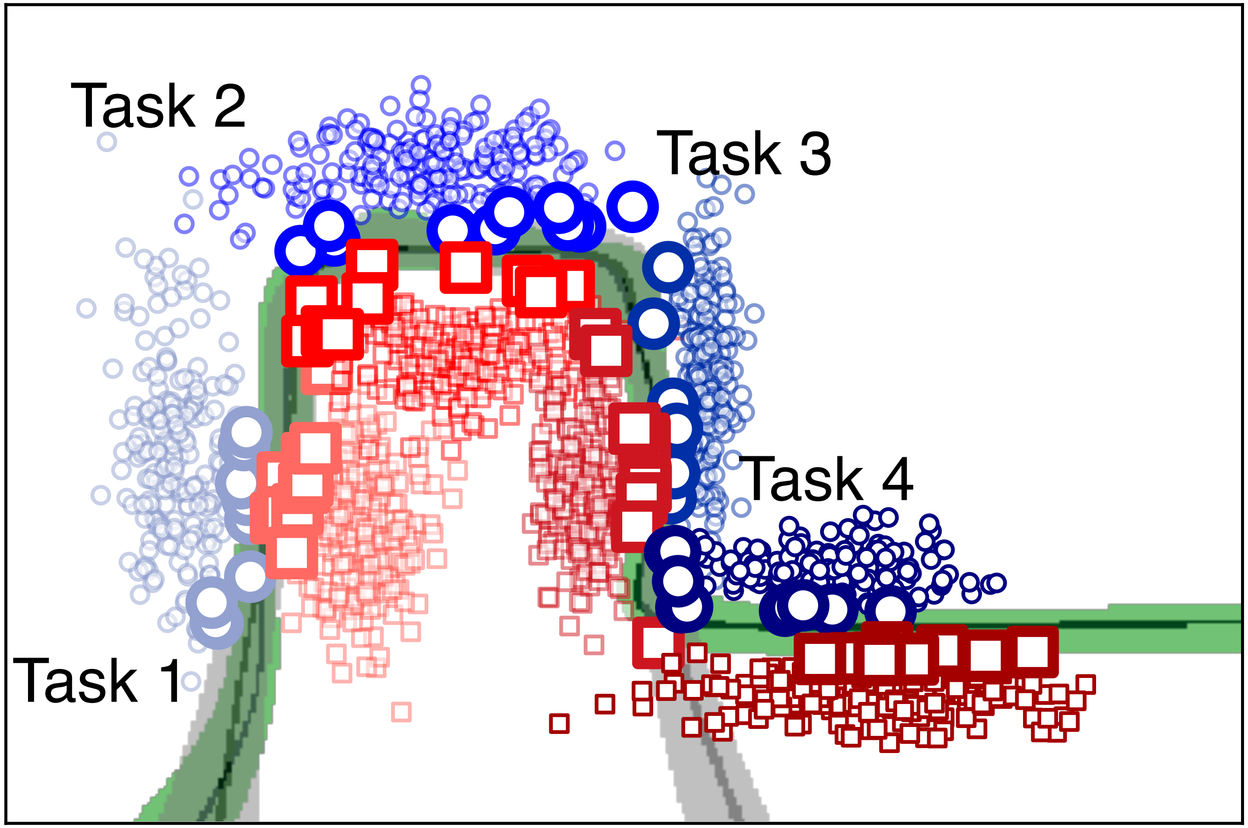
Identify, memorize, and regularize the past obtained using DNN2GP



Pan et al. Continual Deep Learning by Functional Regularisation of Memorable Past, ArXiv 2020

Functional Regularization of Memorable Past (FROMP)

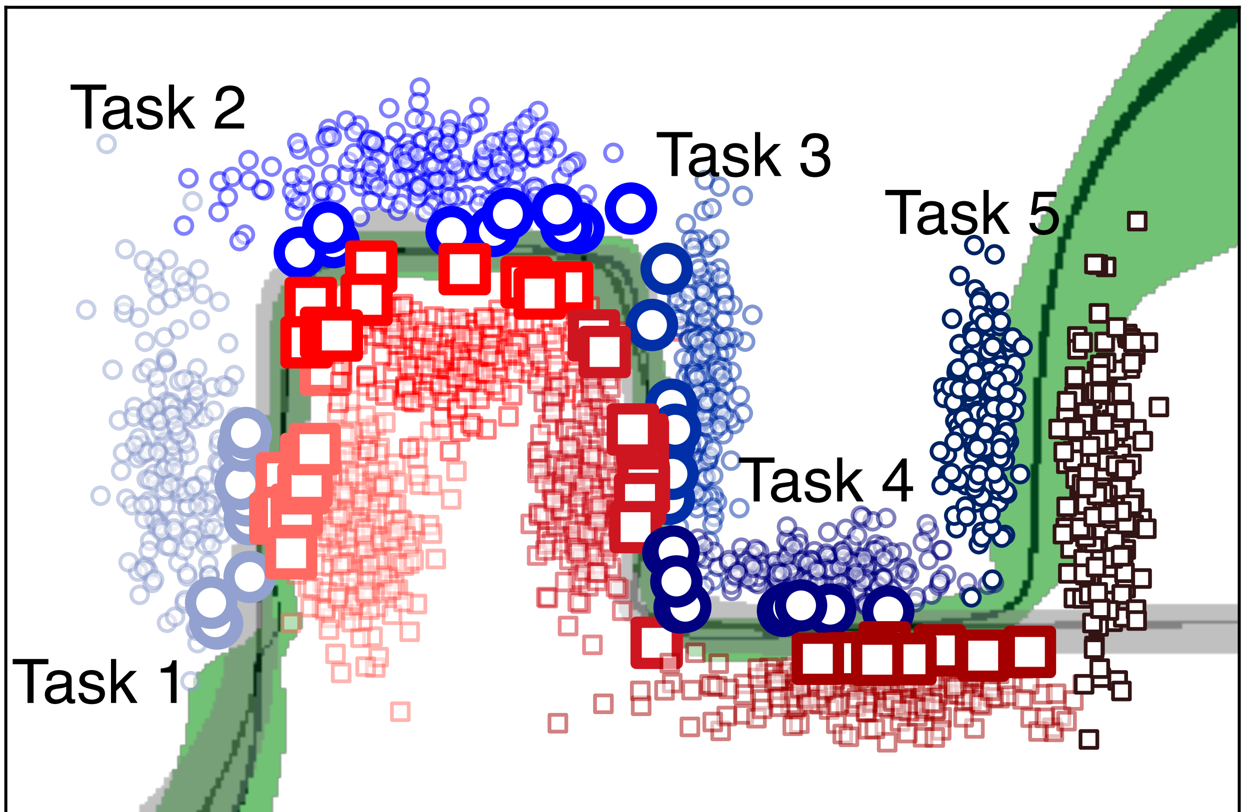
Identify, memorize, and regularize the past obtained using DNN2GP



Pan et al. Continual Deep Learning by Functional Regularisation of Memorable Past, ArXiv 2020

Functional Regularization of Memorable Past (FROMP)

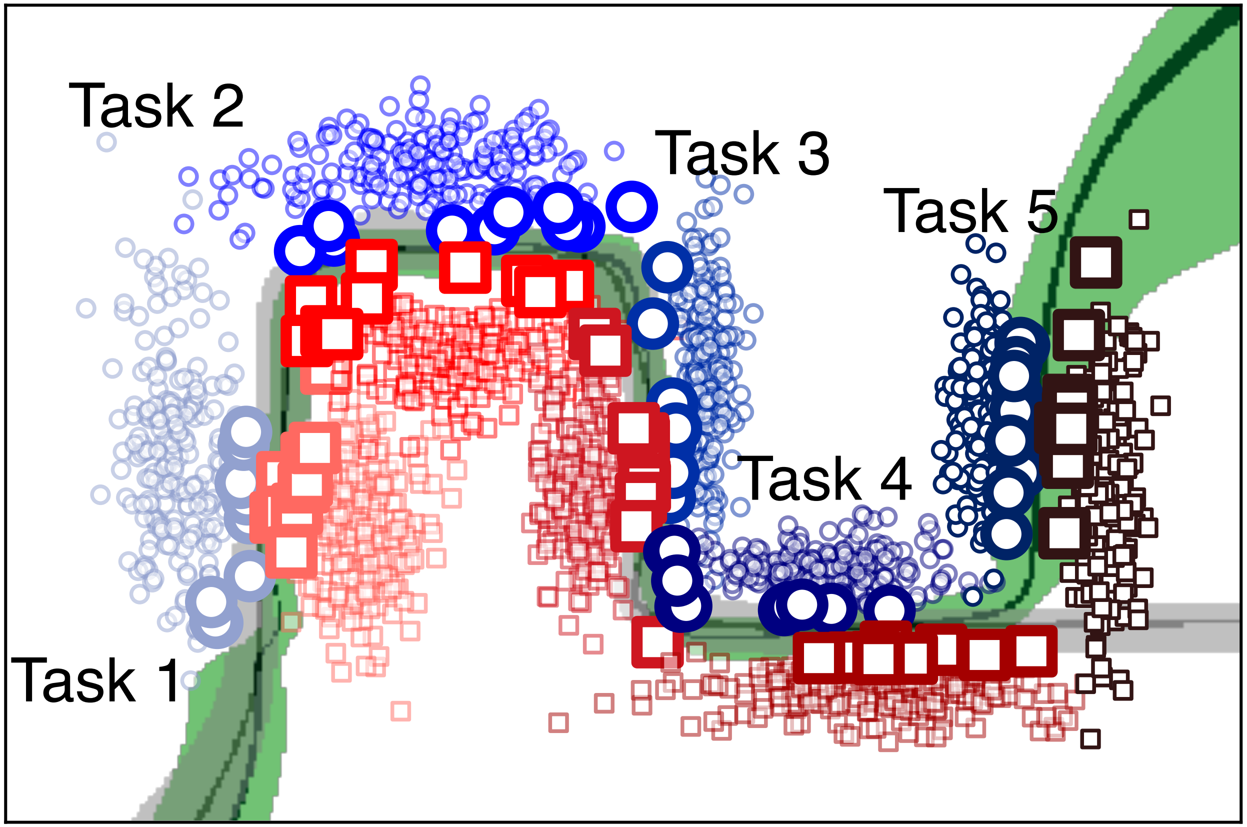
Identify, memorize, and regularize the past obtained using DNN2GP



Pan et al. Continual Deep Learning by Functional Regularisation of Memorable Past, ArXiv 2020

Functional Regularization of Memorable Past (FROMP)

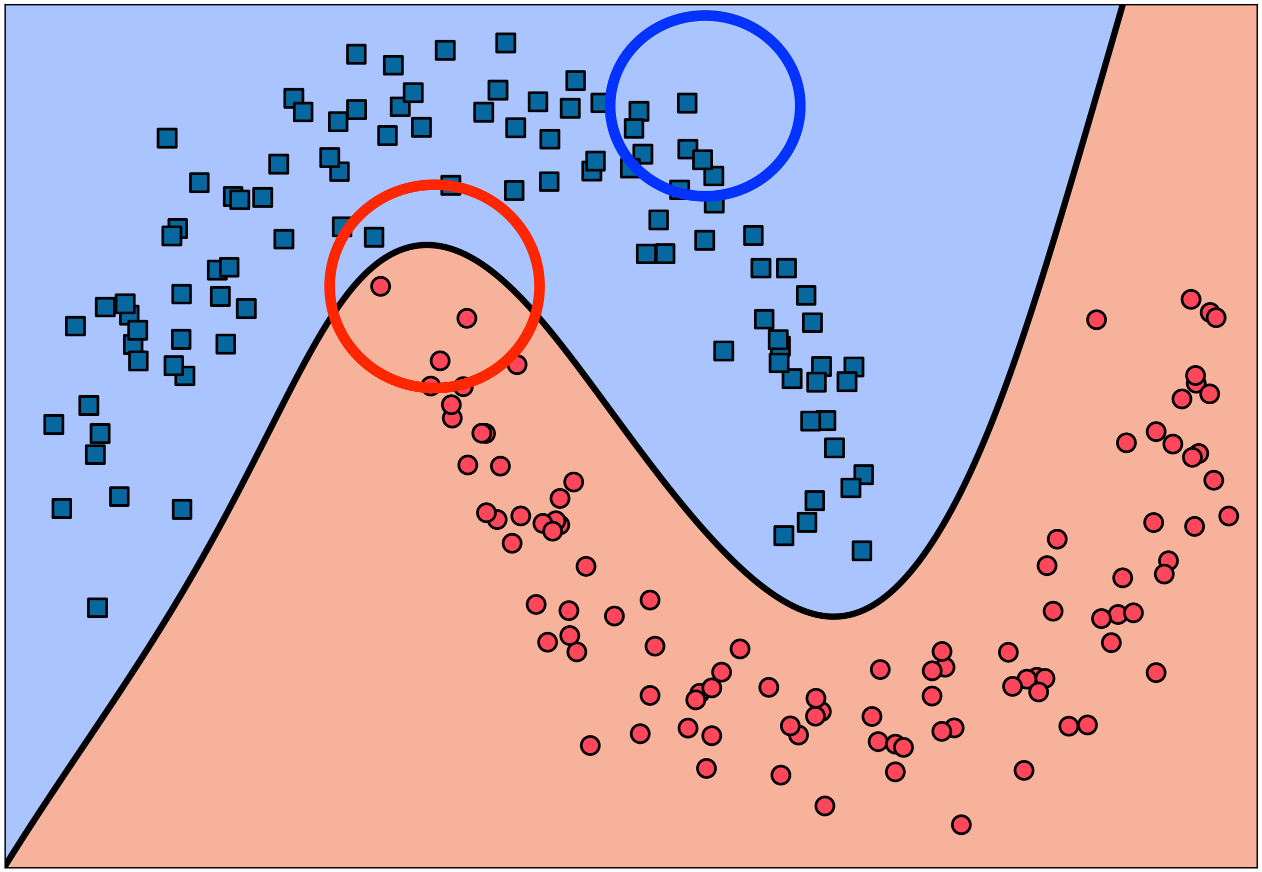
Identify, memorize, and regularize the past obtained using DNN2GP



Pan et al. Continual Deep Learning by Functional Regularisation of Memorable Past, ArXiv 2020

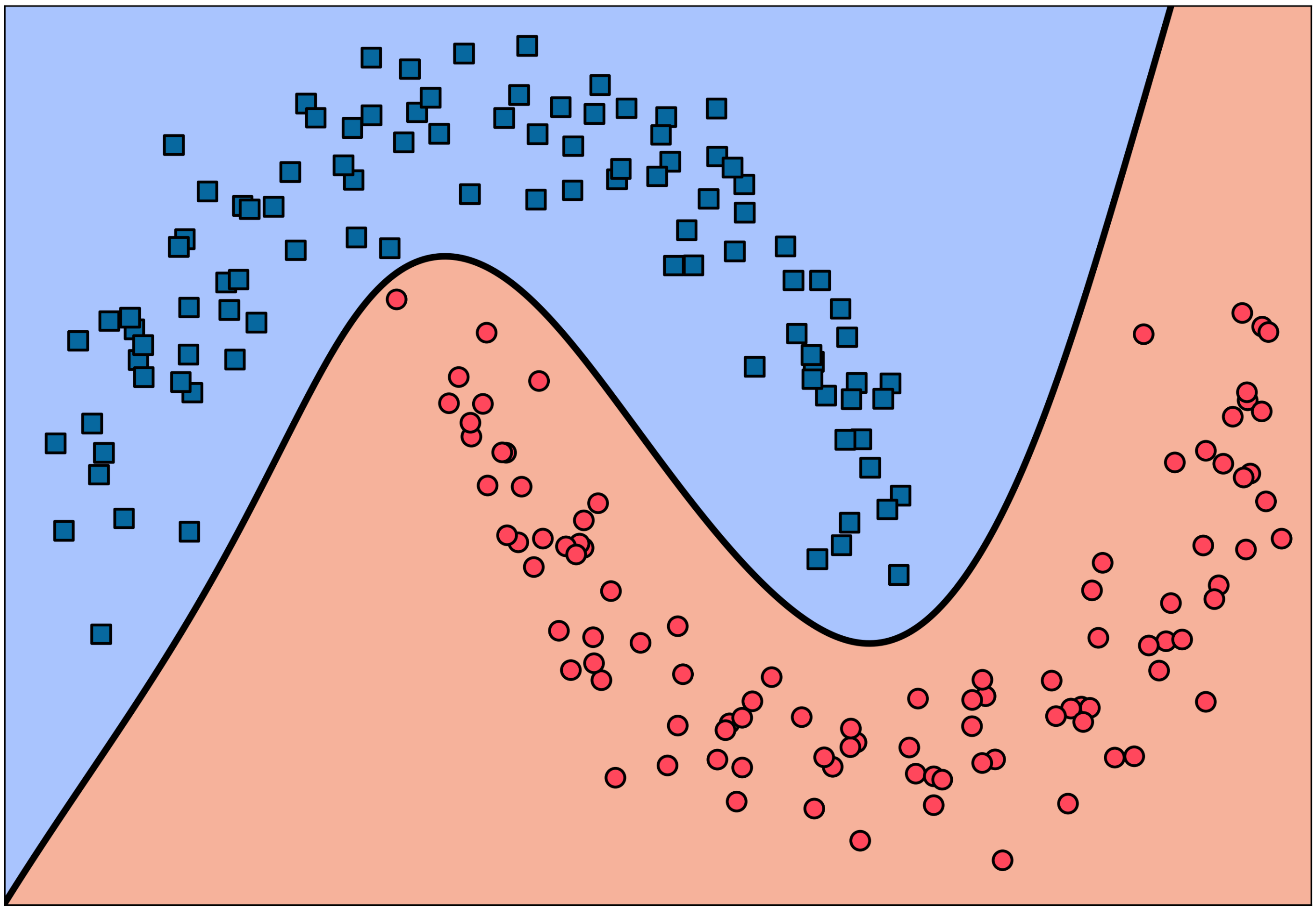
Memorable Past

Which examples are most relevant for the classifier? Red circle vs Blue circle.



Model view vs Data view

DNN2GP provides a measure of relevance

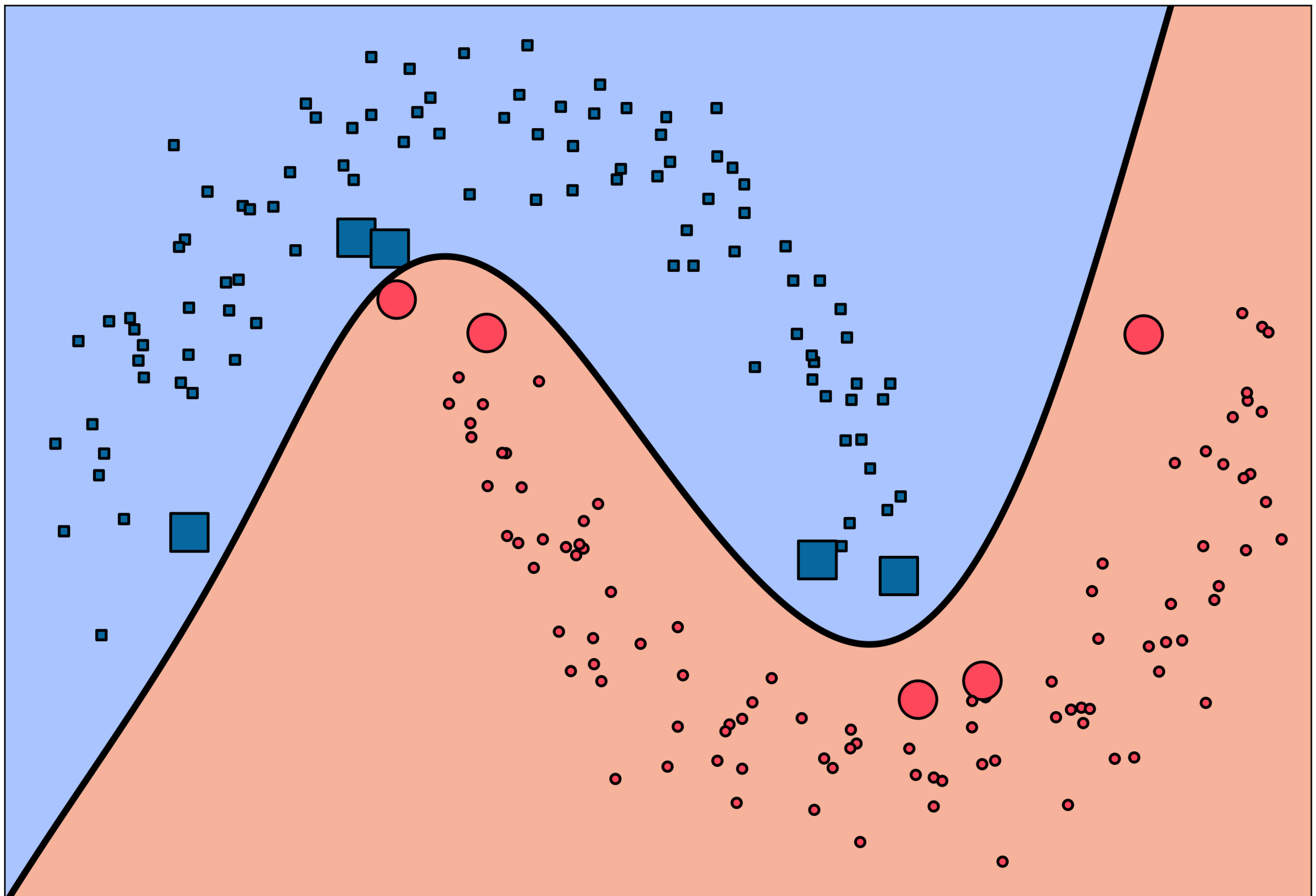


Model view

(By Roman Bachmann)

Model view vs Data view

DNN2GP provides a measure of relevance

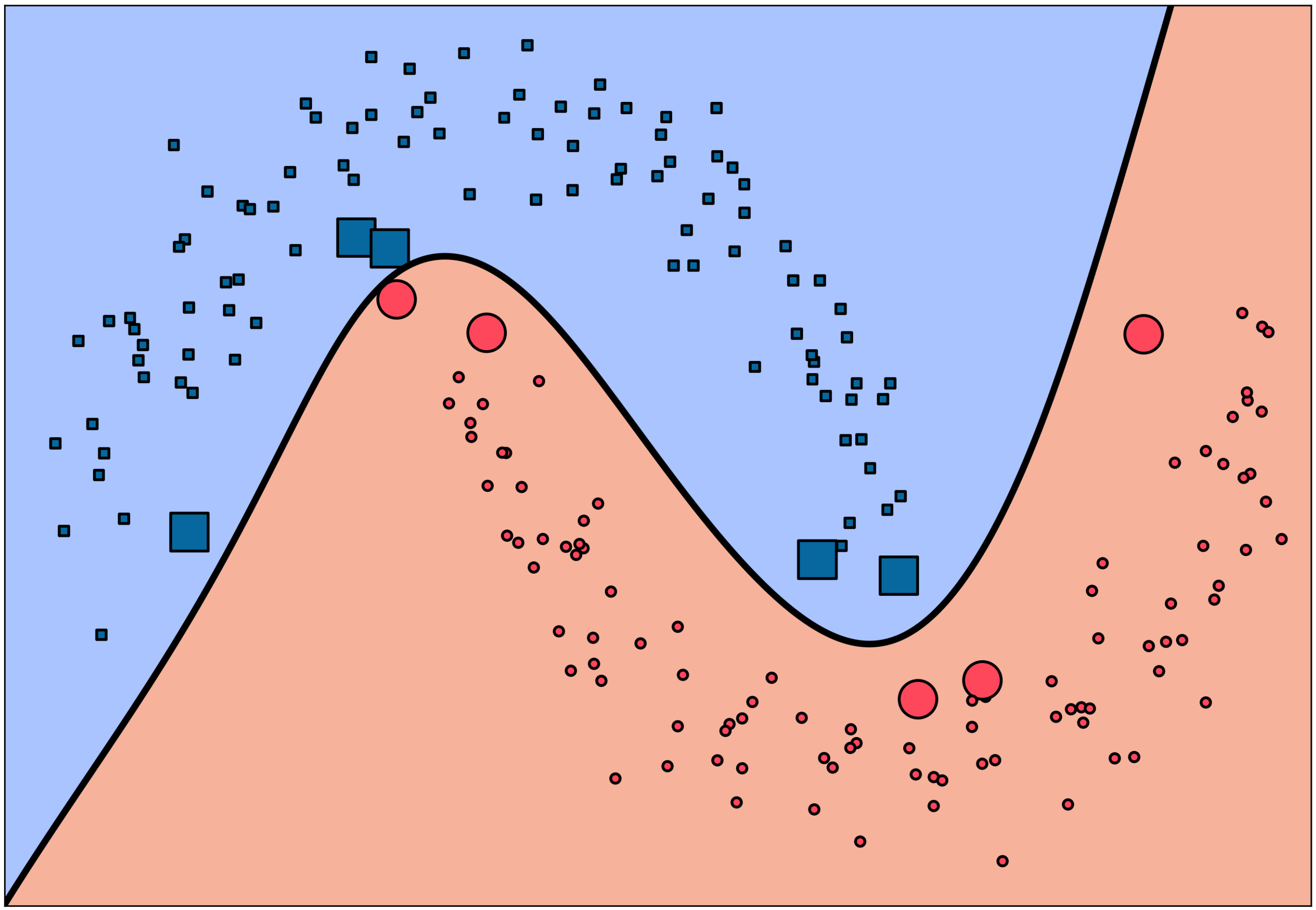


Data view

(By Roman Bachmann)

Model view vs Data view

DNN2GP provides a measure of relevance



Data view

Sort $\Lambda_i(x)$

$= \nabla^2_{ff} \ell(y_i, f_w(x_i))$

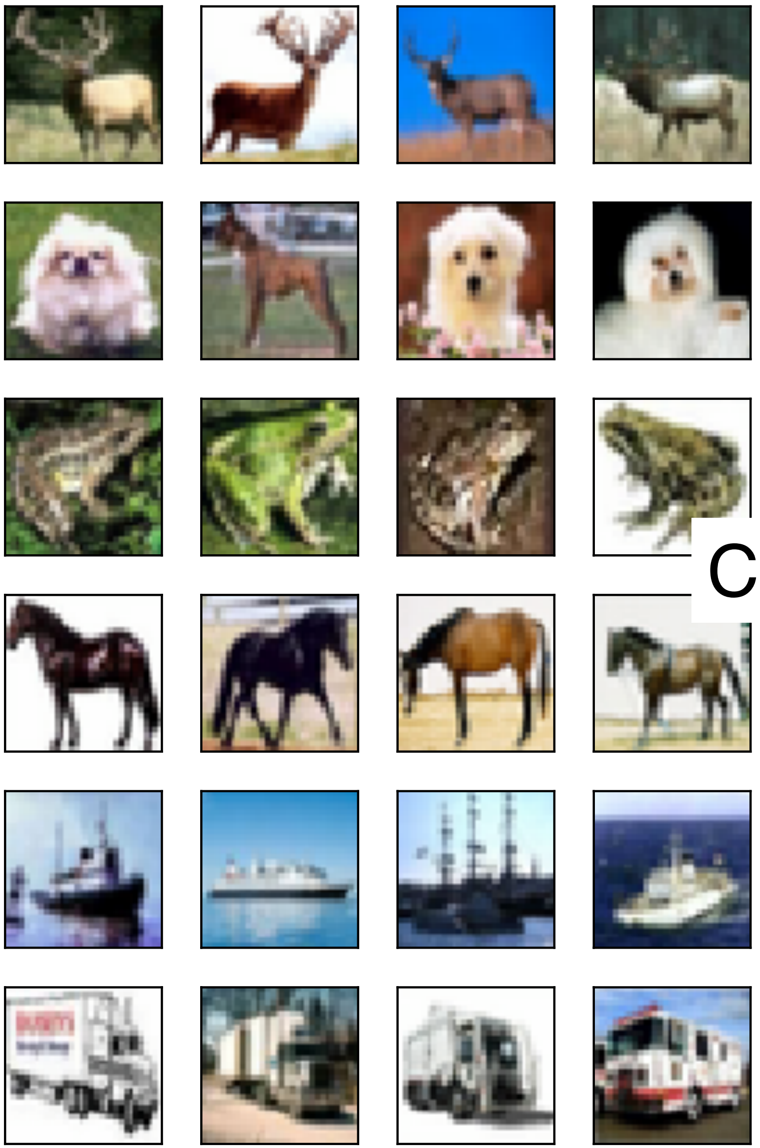
(By Roman Bachmann)

Least Relevant

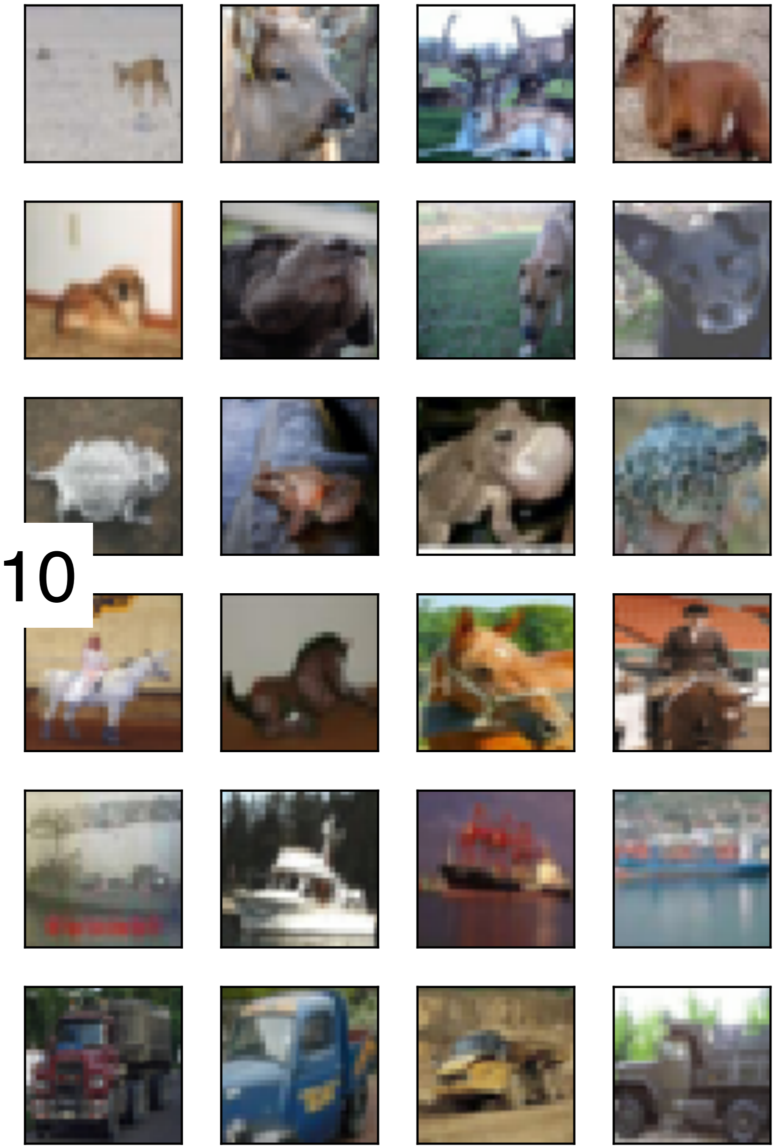
Most Relevant

MNIST

Least Relevant



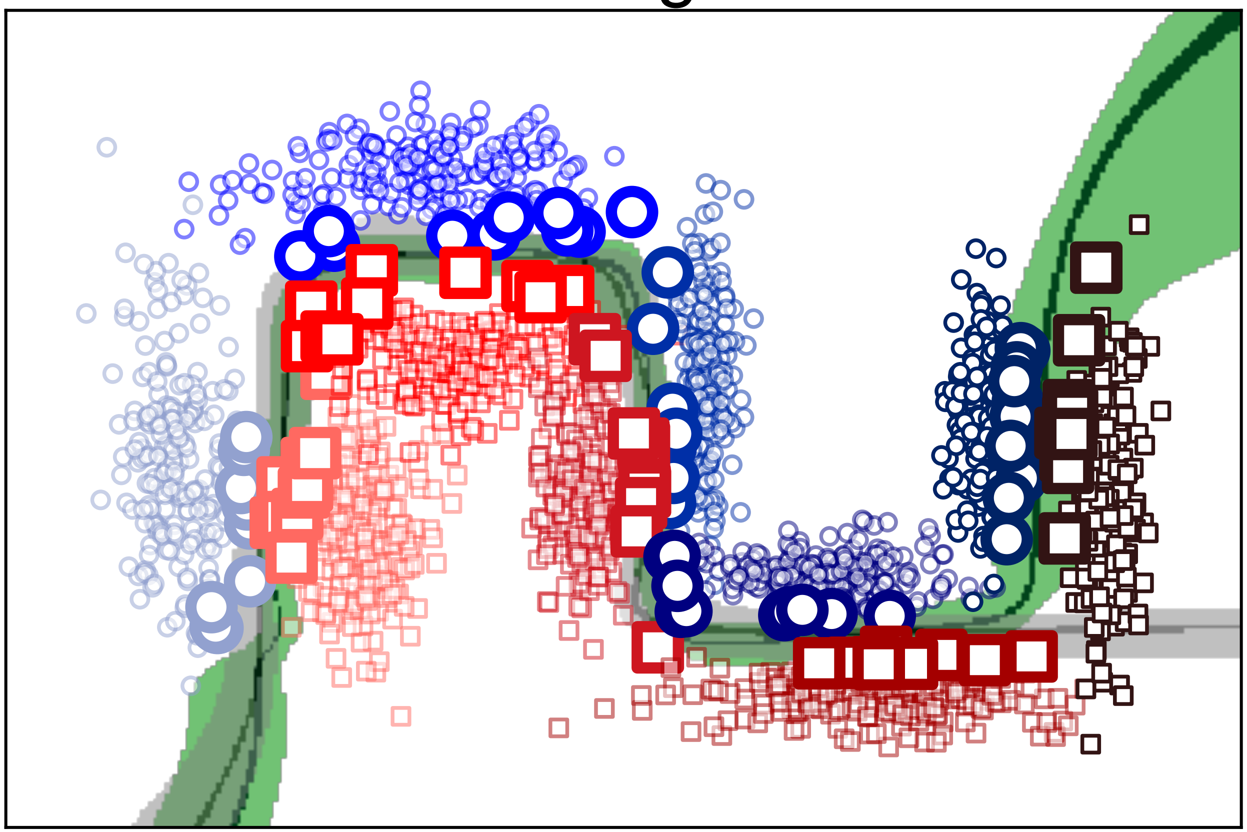
Most Relevant



CIFAR-10

Functional Regularization of Memorable Past (FROMP)

Identify, memorize, and regularize the past obtained using DNN2GP



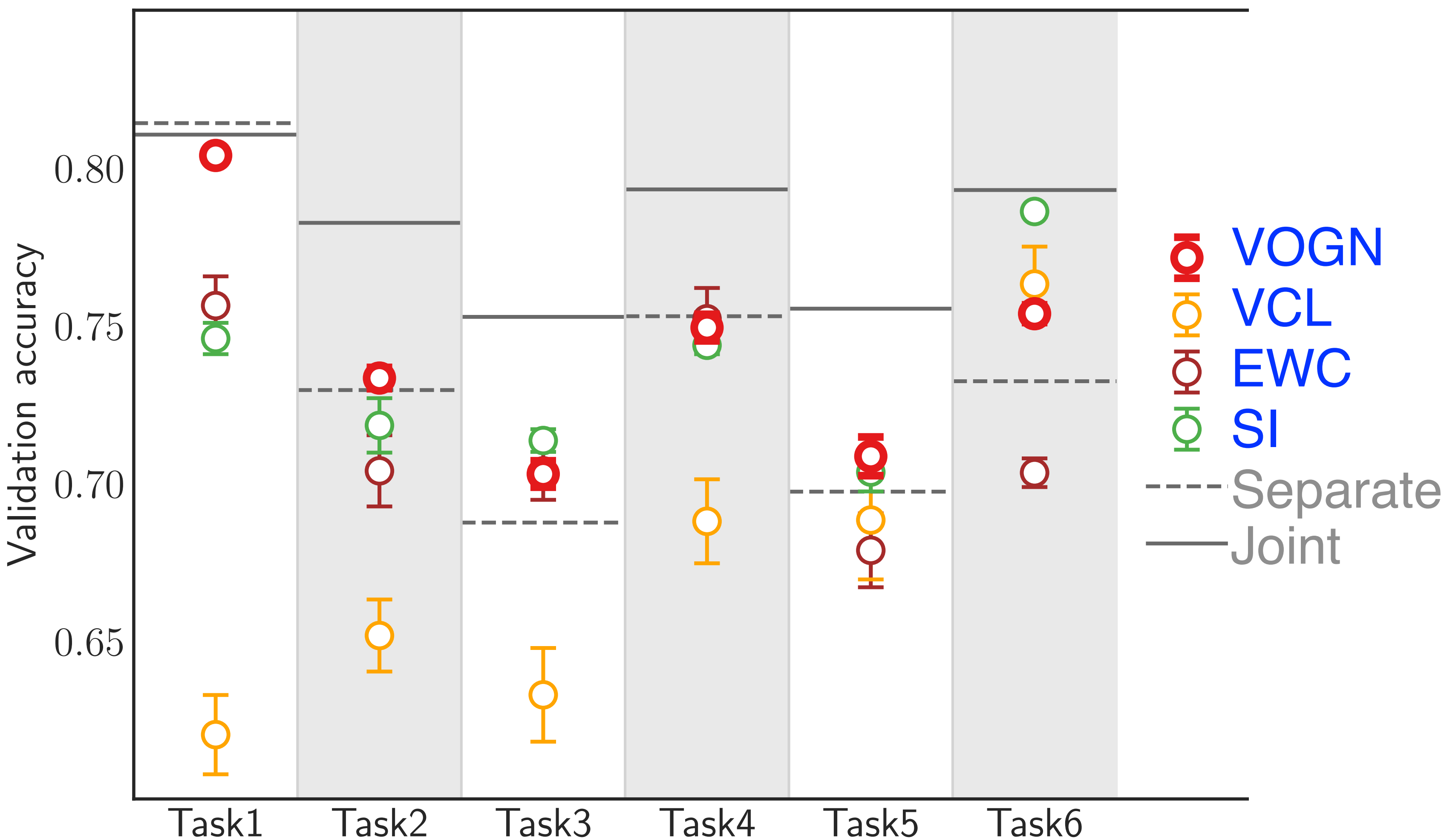
Pan et al. Continual Deep Learning by Functional Regularisation of Memorable Past, ArXiv 2020

(Some) Regularization-based Continual Learning Methods

- Elastic-weight consolidation (EWC) [1]
  - Based on a diagonal Laplace approximation
  - [2] considers structured Laplace
- Synaptic Intelligence (SI) [3]
- Variational Continual learning (VCL) [4]
  - Based on variational inference
- Functional Regularization [5]
- With better approximations, we expect the accuracy to improve, but unfortunately we don't see this!

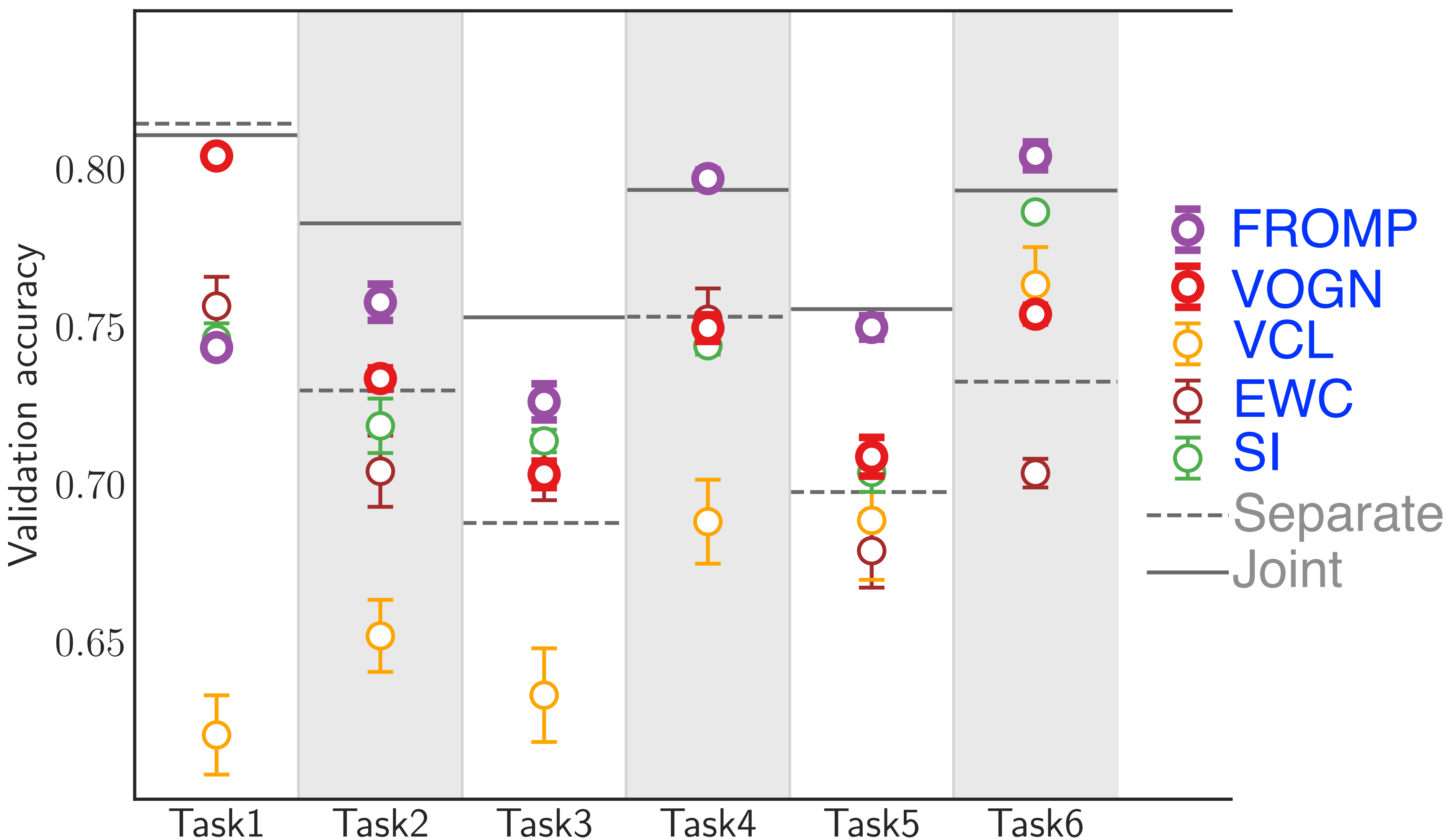
1. Kirkpatrick, James, et al. "Overcoming catastrophic forgetting in neural networks." *PNAS* (2017).
2. Ritter et al. "Online structured laplace … for overcoming catastrophic forgetting." *NeurIPs*. 2018.
3. Zenke et al. "Continual learning through synaptic intelligence." *ICML,* 2017.
4. Nguyen et al. "Variational continual learning." *arXiv preprint arXiv:1710.10628* (2017).
5. Titsias et al. "Functional Regularisation for Continual Learning with Gaussian Processes." *ICLR* (2019).

FROMP improves over EWC!



1. Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks." *PNAS* (2017)

FROMP improves over EWC!



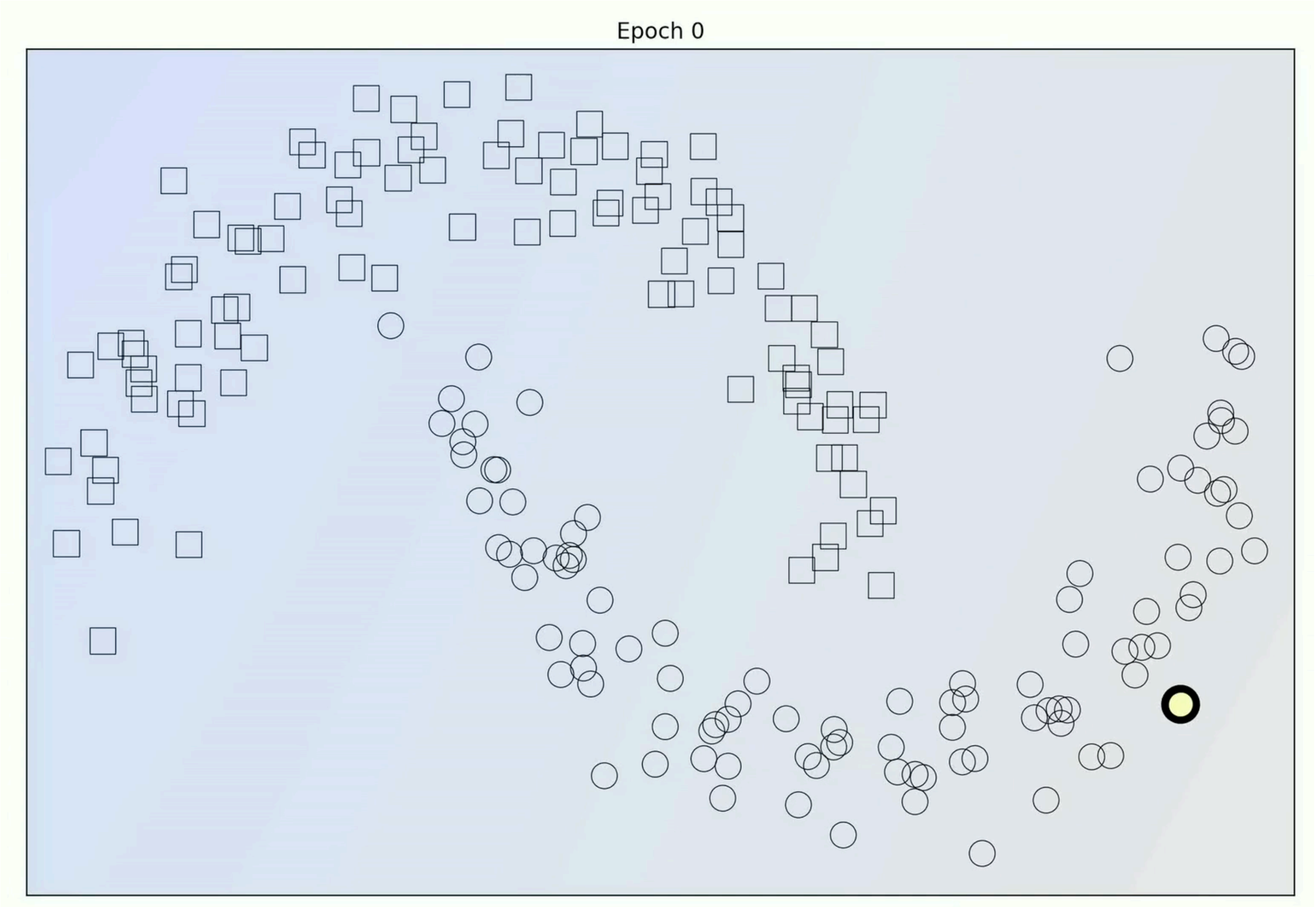
Pan et al. Continual Deep Learning by Functional Regularisation of Memorable Past, ArXiv 2020

Challenges in Life-Long Learning

- Computing exact posterior is not tractable
- Approximations do not always behave the way we want them to
  - They can miss important information from the past and lead to forgetting
- Working with the function space is one solution.
- There are plenty of non-Bayesian solutions, but my personal (biased) opinion is that they are in fact related to Bayesian principles

Active Deep Learning

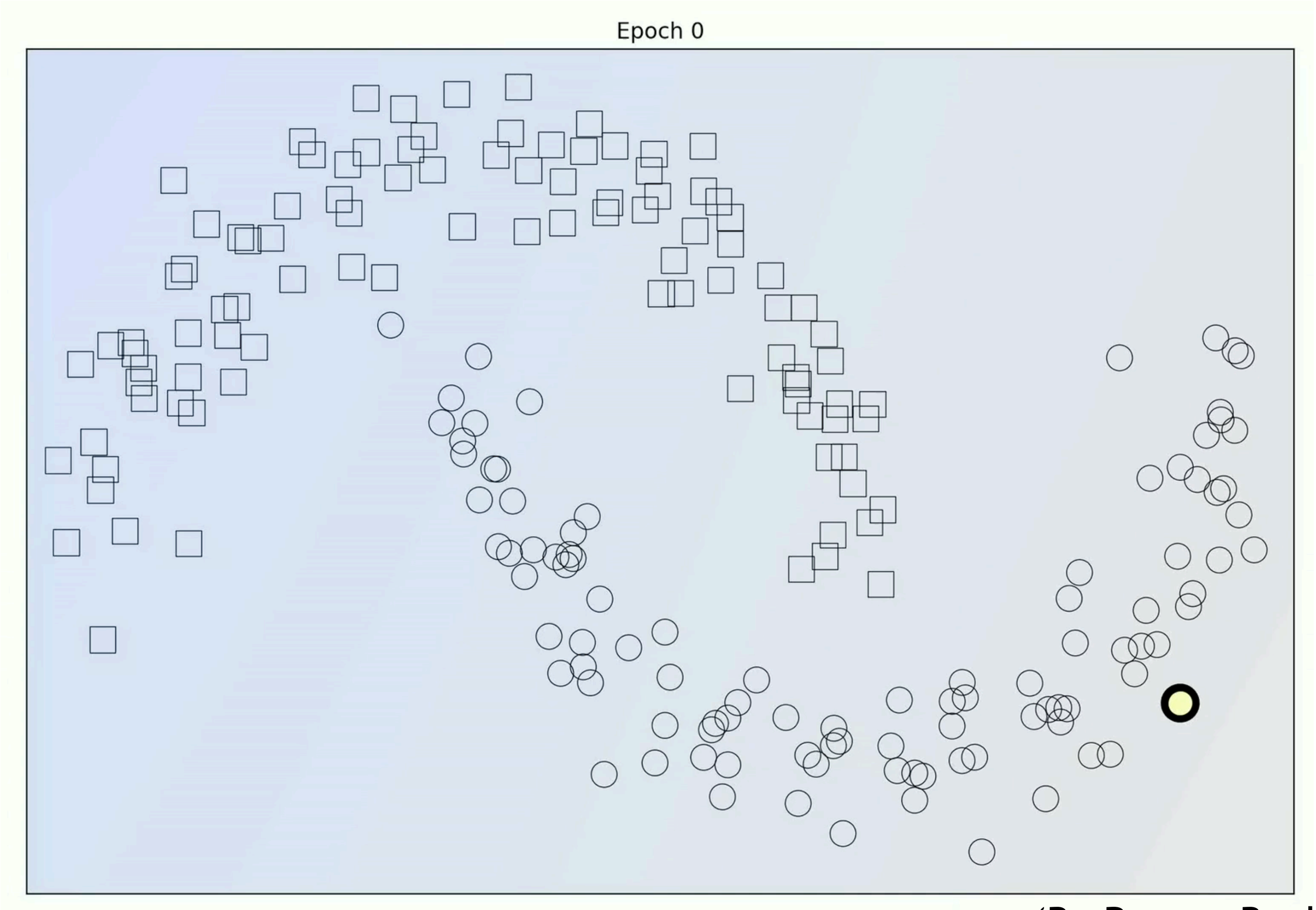
Select "Important" examples while training with Adam



(By Roman Bachmann)

Active Deep Learning

Select “Important” examples while training with Adam



(By Roman Bachmann)

Deep Learning with Bayesian Principles

- Bayesian principles as common principles
  - By computing “posterior approximations”
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and life-long learning
- Impact: Many learning-algorithms with a common set of principles.

Open Challenges

Open Challenges

- How to achieve Life-long deep learning?

Open Challenges

- How to achieve Life-long deep learning?
- How to compute better posterior approx?

Open Challenges

- How to achieve Life-long deep learning?
- How to compute better posterior approx?
- How to compute higher-order gradients?

Towards Life-Long Learning

1. Friston, K. "The free-energy principle: a unified brain theory?." *Nature neuroscience* (2010)

Towards Life-Long Learning

- Three questions
  - Q1: What do we know? (model)
  - Q2: What do we not know? (uncertainty)
  - Q3: What do we need to know? (action & exploration)

1. Friston, K. "The free-energy principle: a unified brain theory?." *Nature neuroscience* (2010)

Towards Life-Long Learning

- Three questions
  - Q1: What do we know? (model)
  - Q2: What do we not know? (uncertainty)
  - Q3: What do we need to know? (action & exploration)
- Posterior approximation is a key element
  - Models == representation of the world
  - Approximations == representation of the model
  - Improve the model through actions collect more data (act to appropriately "fill" the data space)

1. Friston, K. "The free-energy principle: a unified brain theory?." *Nature neuroscience* (2010)

Learning-Algorithms from Bayesian Principles

Coming soon!

A preliminary version is at

https://emtiyaz.github.io/papers/learning_from_bayes.pdf

Havard Rue (KAUST)

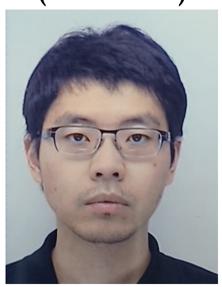
Acknowledgements

Slides, papers, & code are at emtiyaz.github.io

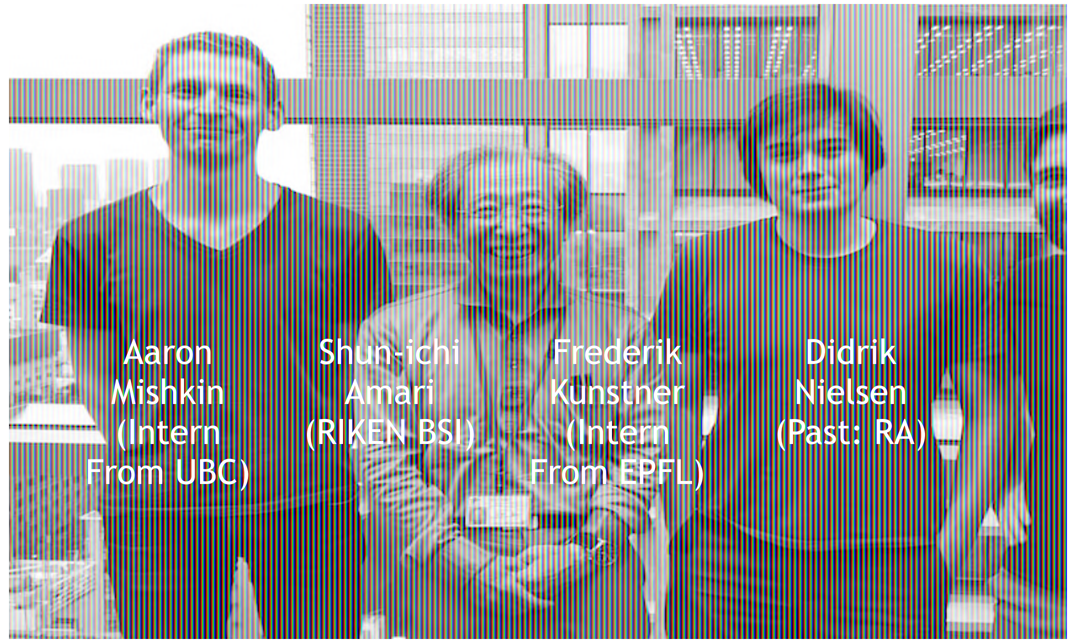
Wu Lin
(Past: RA)

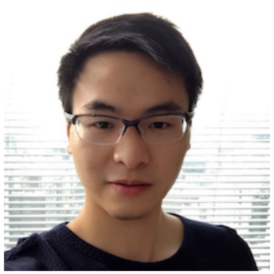
Nicolas Hubacher
(Past: RA)

Masashi Sugiyama
(Director RIKEN-AIP)



Voot Tangkaratt
(Postdoc, RIKEN-AIP)





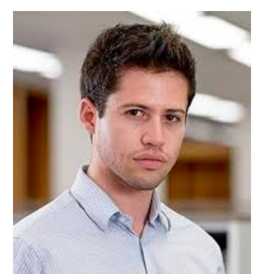
Zuozhu Liu
(Intern from SUTD)

RAIDEN

External Collaborators

Mark Schmidt
(UBC)

Reza Babanezhad
(UBC)



Yarin Gal
(UOxford)

Akash Srivastava
(UEdinburgh)

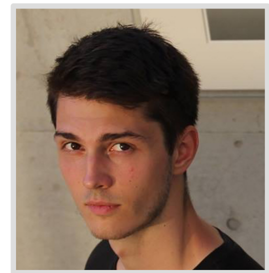
Acknowledgements

Slides, papers, & code
are at emtiyaz.github.io

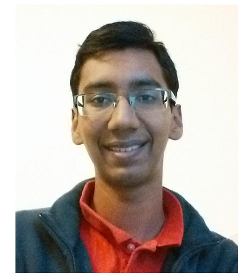
Kazuki Osawa
(Tokyo Tech)

Rio Yokota
(Tokyo Tech)

Anirudh Jain
(Intern from
IIT-ISM, India)



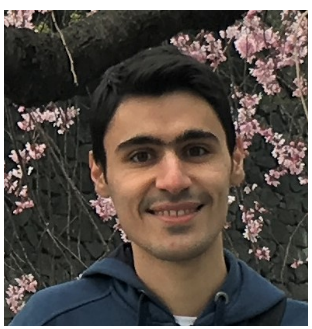
Runa Eschenhagen
(Intern from
University of
Osnabruck)



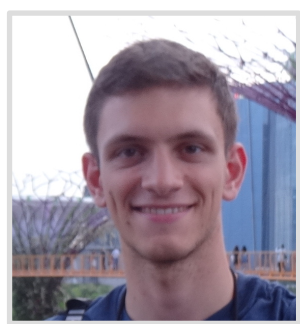
Siddharth
Swaroop
(University of
Cambridge)

Rich Turner
(University of
Cambridge)

Alexander Immer
(Intern from EPFL)



Ehsan Abedi
(Intern from EPFL)



Maciej Korzepa
(Intern from DTU)

Pierre Alquier
(RIKEN AIP)

Havard Rue
(KAUST)

PingBo Pan
(UT Sydney)

Approximate Bayesian Inference Team