Learning-Algorithms from Bayesian Principles

Mohammad Emtiyaz Khan
RIKEN Center for AI Project, Tokyo
http://emtiyaz.github.io

The Goal of My Research

"To understand 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.

Converged at the age of 12 months

Transfer Knowledge at the age of 14 months

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

"Continual" learning of incremental information from non-stationary data

"Bulk" learning of all possible information from stationary data

My current research focuses on reducing this gap!

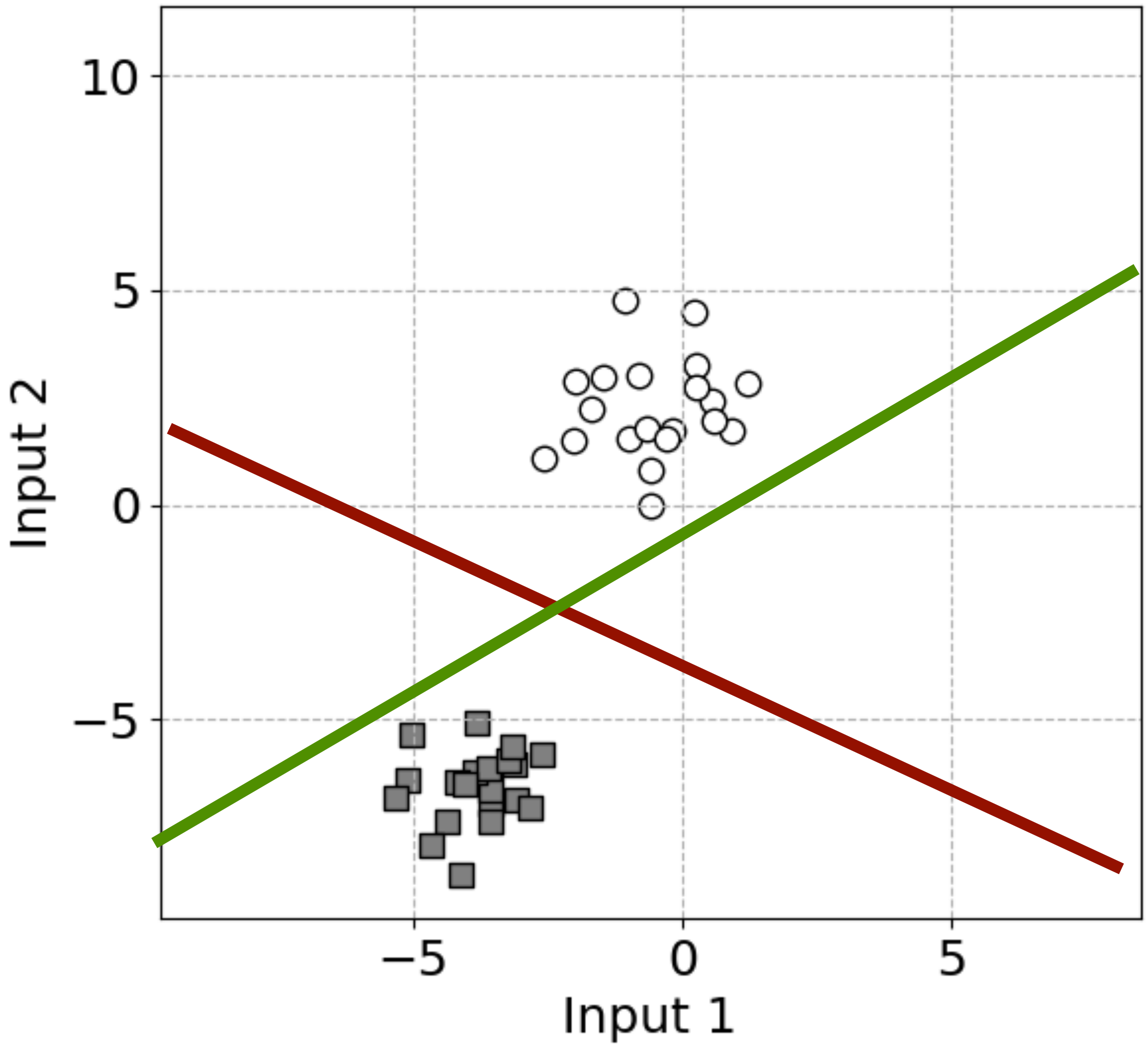
Continual lifelong learning with NN (Parisi et al. 2019)

Learning-Algorithms from Bayesian Principles

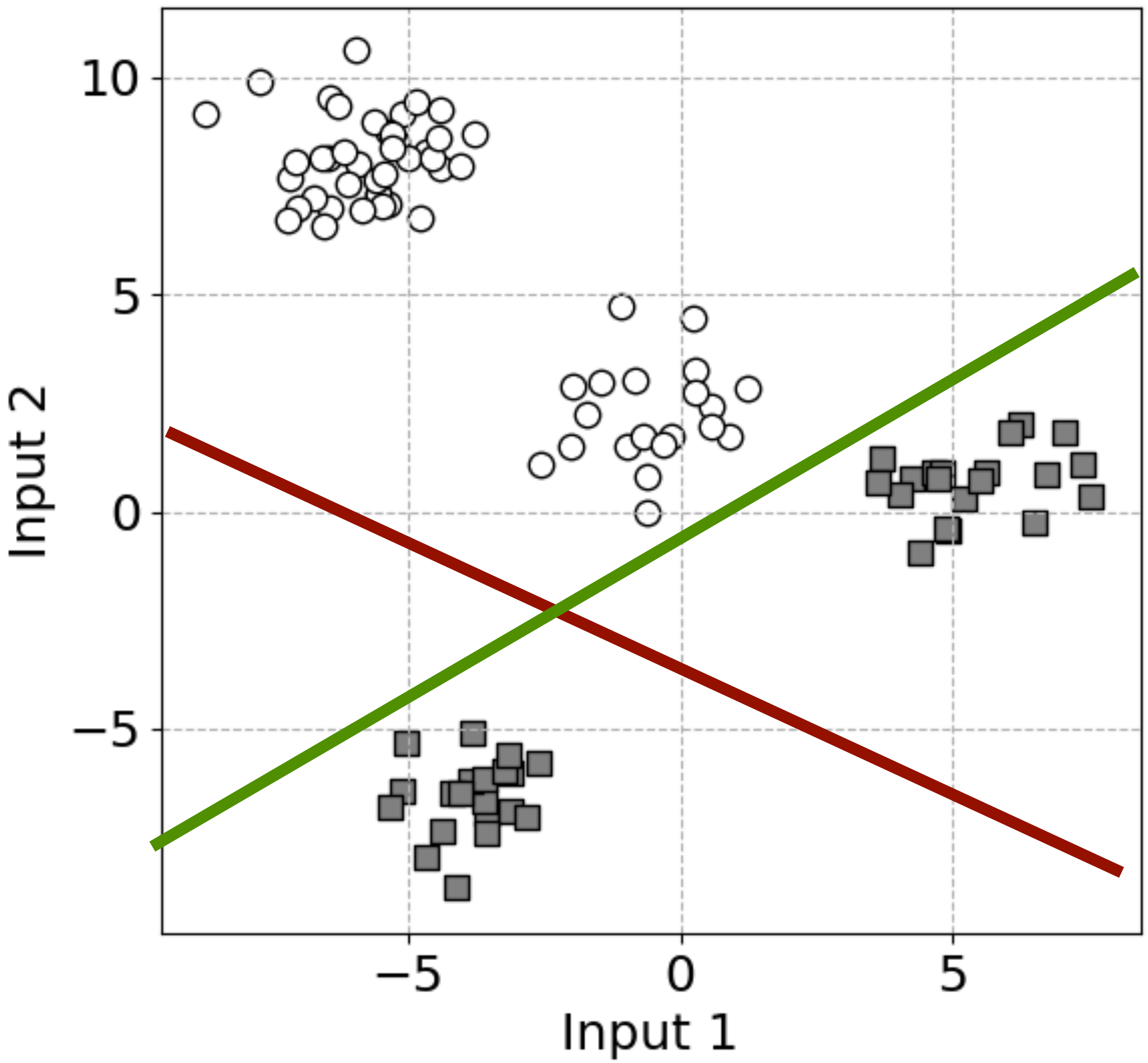
- Practical Bayesian principles.
- Bayesian learning rule
  - a generalization of many learning-algorithms,
    - Classical (least-squares, Newton, HMM, Kalman.. etc).
    - Deep Learning (SGD, RMSprop, Adam).
- Data relevance
- Continual Learning with Bayes
- Impact: Everything with one common principle.

Why Bayes?

Which is a good classifier?

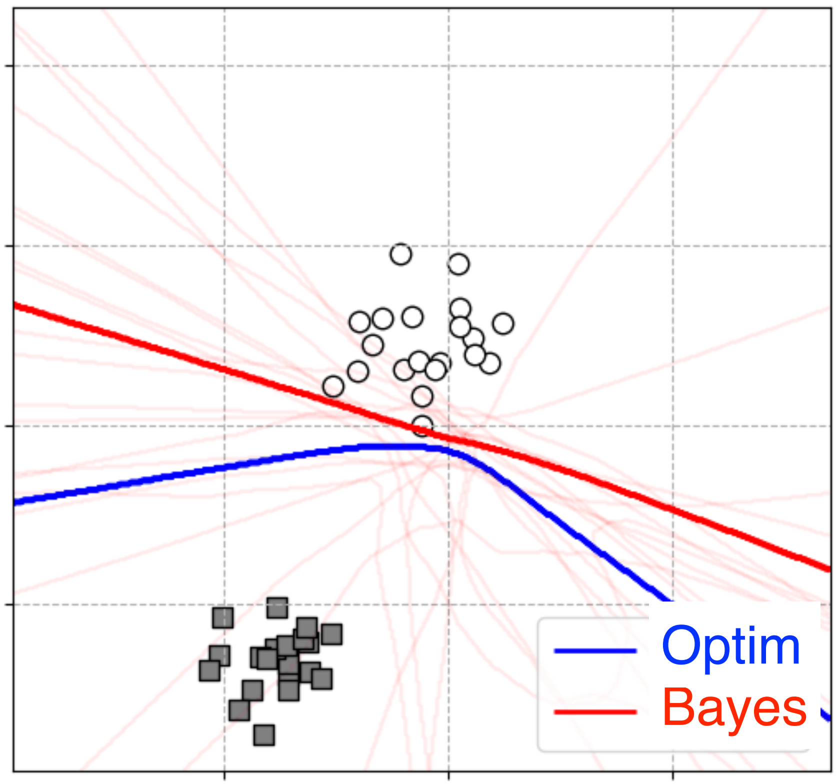


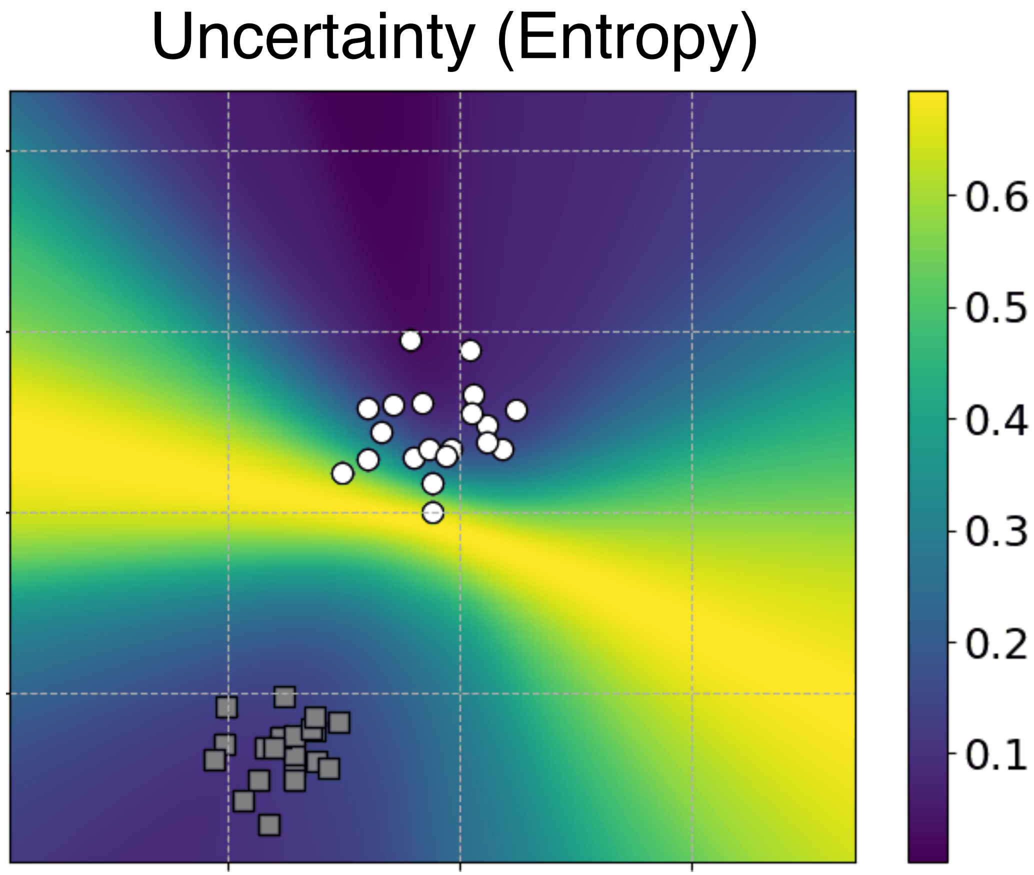
Which is a good classifier?



"What the model does not know but should know": Knowledge gap

The Bayesian Solution





(By Kazuki Osawa) https://github.com/team-approx-bayes/dl-with-bayes

Optimization -> Bayes

NeurIPS 2019

Switching from “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

Bayesian Learning Rule

Learning algorithms from Bayes

Uncertainty for free

Data relevance for free

Bayes Rule as Optimization

Estimate a distribution over model parameters.

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

Optimization formulation $\quad \ell(\mathcal{D},\theta) := \log p(\mathcal{D}|\theta)p(\theta)$

Distribution (e.g. Gaussian)

Entropy

$$\max_{\lambda} \; -\mathbb{E}_{q_\lambda(\theta)}[\ell(\mathcal{D},\theta)] - \mathcal{H}(q)$$

Exploitation

Exploration

Parameters
(e.g., mean and variance)

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

Learning-Algorithms by Bayesian Principles

AIstats 2017

Learning by optimization: $\theta \leftarrow \theta - \rho H^{-1} \nabla_\theta \ell(\theta)$

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

Natural and Expectation parameters of q

e.g., Gaussian distribution

Natural parameters $\{V^{-1}m, V^{-1}\}$

$$q(\theta) := \mathcal{N}(\theta|m, V)$$

Expectation/moment/ mean parameters $\{\mathbb{E}(\theta), \mathbb{E}(\theta\theta^\top)\}$

$$\exp\left[m^\top V^{-1}\theta - \frac{1}{2}\theta^\top V^{-1}\theta\right]$$

Learning by Bayes

Learning by optimization: $\theta \leftarrow \theta - \rho H^{-1} \nabla_\theta \ell(\theta)$

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

Natural and Expectation parameters of q

AIstats 2017
ICML 2017

- Classical algorithms: Least-squares, Newton's method, Kalman filters, Baum-Welch, Forward-backward, etc.
- Bayesian inference: EM, Laplace's method, SVI, VMP.

ICML 2018
NeurIPS 2018
ISITA 2018
ICLR 2018

- Deep learning: SGD, RMSprop, Adam.
- Reinforcement learning: parameter-space exploration, natural policy-search.
- Continual learning: Elastic-weight consolidation.
- Online learning: Exponential-weight average.

NIPS 2017

- Global optimization: Natural evolutionary strategies, Gaussian homotopy, continuation method & smoothed optimization.
- List incomplete…

$q_\lambda(\theta) := \mathcal{N}(m, V)$

Least Squares

$$\lambda \leftarrow (1-\rho)\lambda - \rho \nabla_{\mu} \mathbb{E}_q [\ell(\theta)] \qquad \Rightarrow \lambda_* = \nabla_{\mu_*} \mathbb{E}_{q_*} [\ell(\theta)]$$

$$\mathbb{E}_q \left(\underbrace{\overbrace{(y - X\theta)^\top (y - X\theta)}^{\text{likelihood}}}_{-\mathbb{E}_{q_\lambda}[\theta]^\top X^\top y + \text{trace}\left[X^\top X \mathbb{E}_{q_\lambda}[\theta\theta^\top]\right]} + \overbrace{\gamma\theta^\top\theta}^{\text{prior}} \right) := \ell(\theta)$$

$$\begin{aligned} \nabla_{\mathbb{E}_{q_\lambda}[\theta]} &= \\ \nabla_{\mathbb{E}_{q_\lambda}[\theta\theta^\top]} &= \end{aligned} \begin{pmatrix} -X^\top y & + & 0 \\ X^\top X & + & \gamma I \end{pmatrix} \begin{aligned} &= V^{-1} m \\ &= V^{-1} \end{aligned}$$

Expectation params

$$[X^\top X + \gamma I]^{-1} X^\top y$$

Learning by Bayes for DNNs

ICML 2018

$$\mathbb{E}_q \left[\sum_{i=1}^{N} \underbrace{\overbrace{\ell(y_i, f_\theta(x_i))}^{\text{likelihood}}}_{\text{neural network}} + \overbrace{\gamma \theta^\top \theta}^{\text{prior}} \right]$$

$$q(\theta) := \mathcal{N}(\theta | m, \mathrm{Diag}(v))$$

RMSprop

$$\theta \leftarrow \mu$$
$$g \leftarrow \tfrac{1}{M} \sum_i \nabla_\theta \, \ell(y_i, f_\theta(x_i))$$
$$s \leftarrow (1-\beta)s + \beta g^2$$
$$\mu \leftarrow \mu + \alpha \, \frac{g}{\sqrt{s} + \delta}$$

Bayes with diagonal Gaussian

$$\theta \leftarrow \mu + \epsilon, \;\; \text{where } \epsilon \sim \mathcal{N}(0, Ns + \gamma)$$
$$g \leftarrow \tfrac{1}{M} \sum_i \nabla_\theta \, \ell(y_i, f_\theta(x_i))$$
$$s \leftarrow (1-\beta)s + \beta \tfrac{1}{M} \sum_i [\nabla_\theta \ell(y_i, f_\theta(x_i))]^2$$
$$\mu \leftarrow \mu + \alpha \, \frac{g + \gamma\mu/N}{s + \gamma/N}$$

Optimization -> Bayes

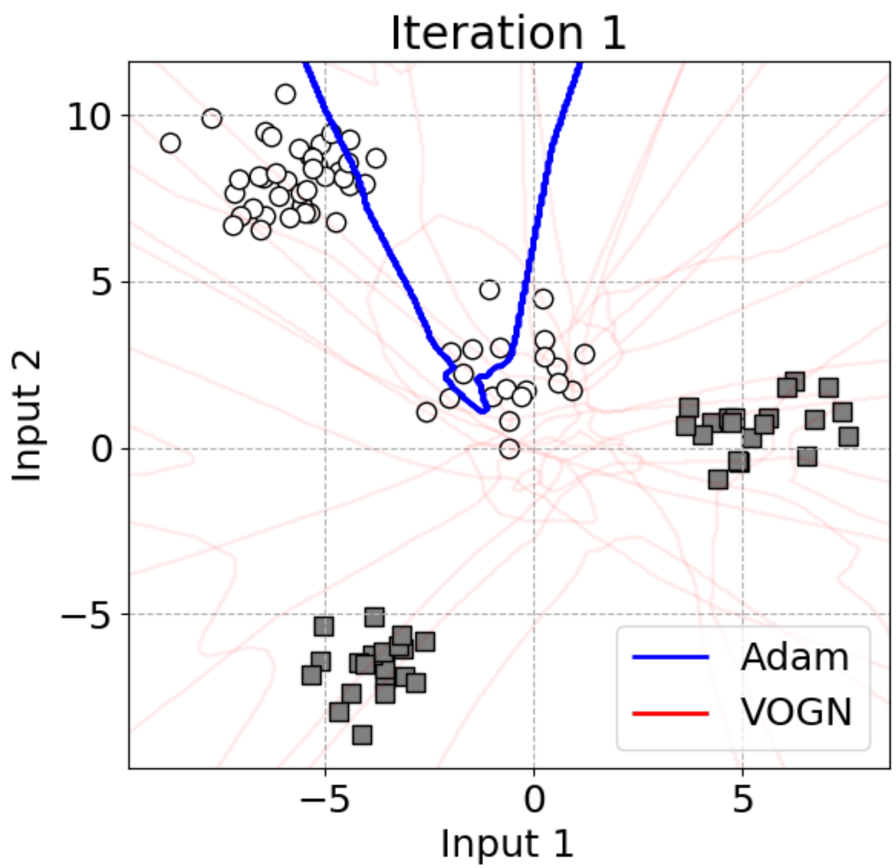
NeurIPS 2019

Switching from “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))

for data, target in train_loader:

    def closure():
        optimizer.zero_grad()
        output = model(data)
        loss = F.binary_cross_entropy_with_logits(output, target)
        loss.backward()
        return loss, output

    loss, output = optimizer.step(closure)
```

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

Fast Uncertainty in Deep Learning



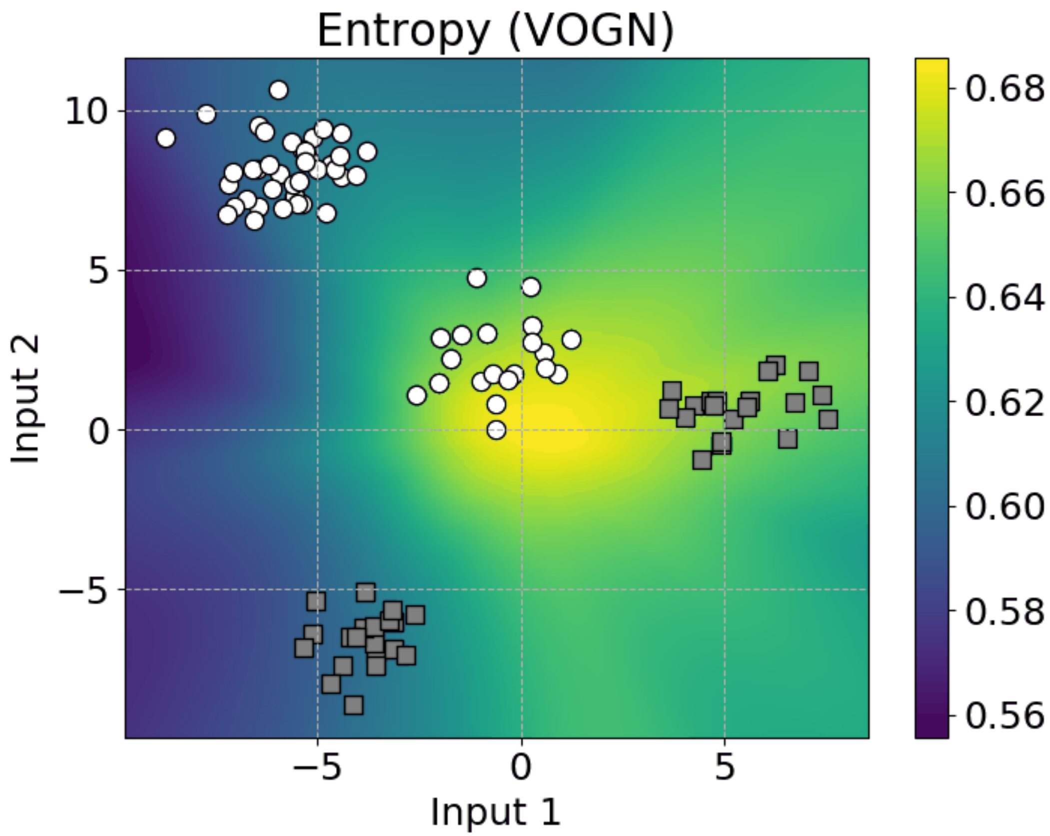
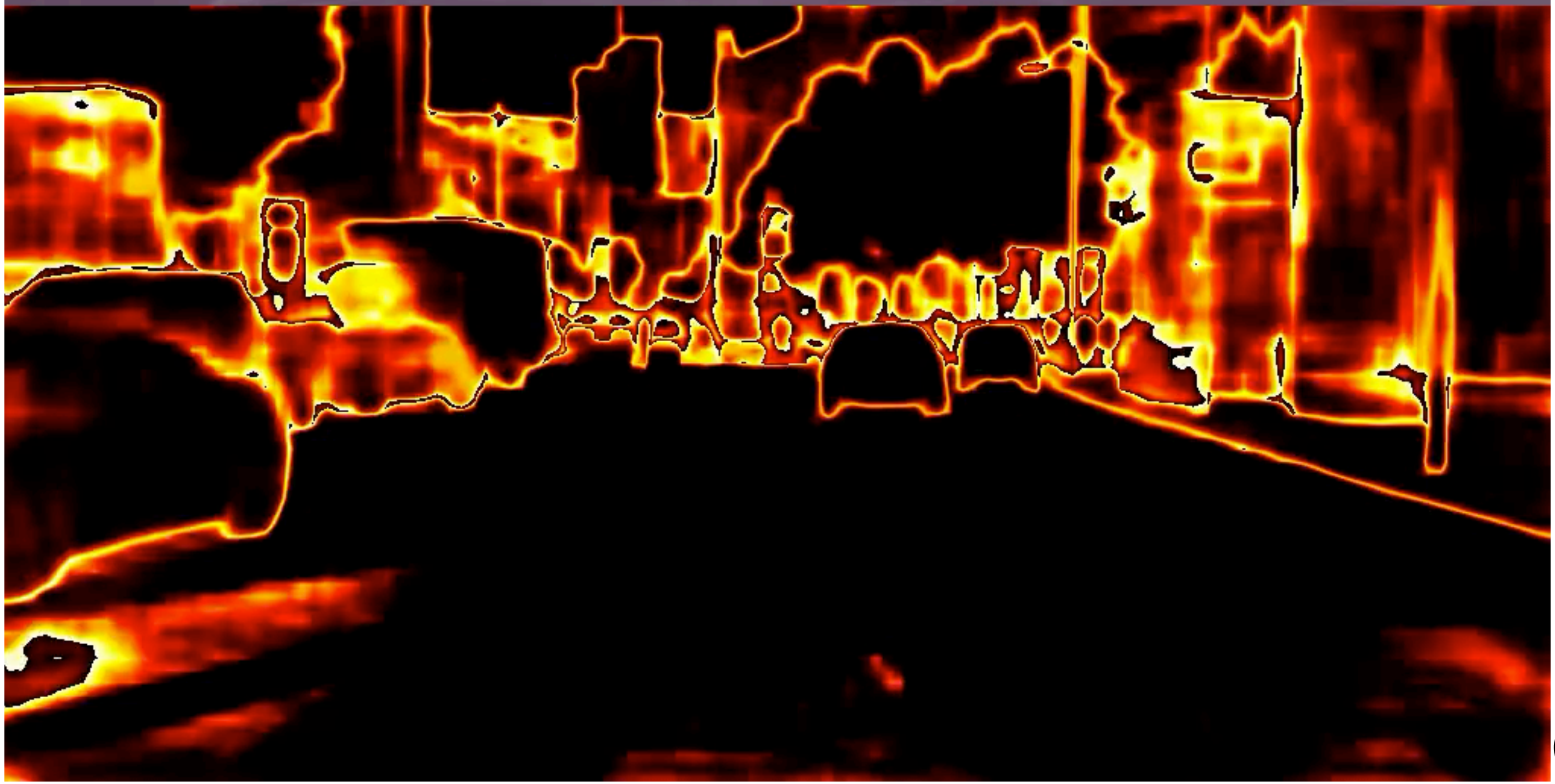


Image Segmentation



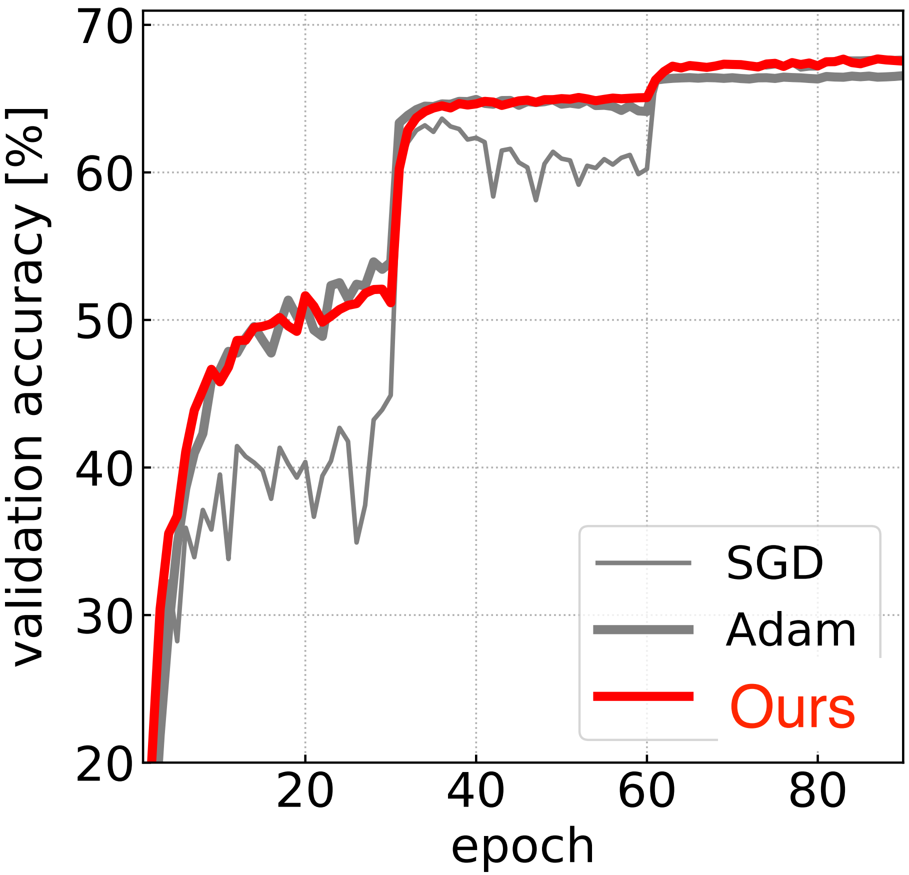
Uncertainty (entropy of class probs)

(By Roman Bachmann)

Practical DL with Bayes (on ImageNet)

State-of-the-art performance and convergence rate, while preserving benefits of Bayesian principles ("well-calibrated" uncertainty).

Accuracy

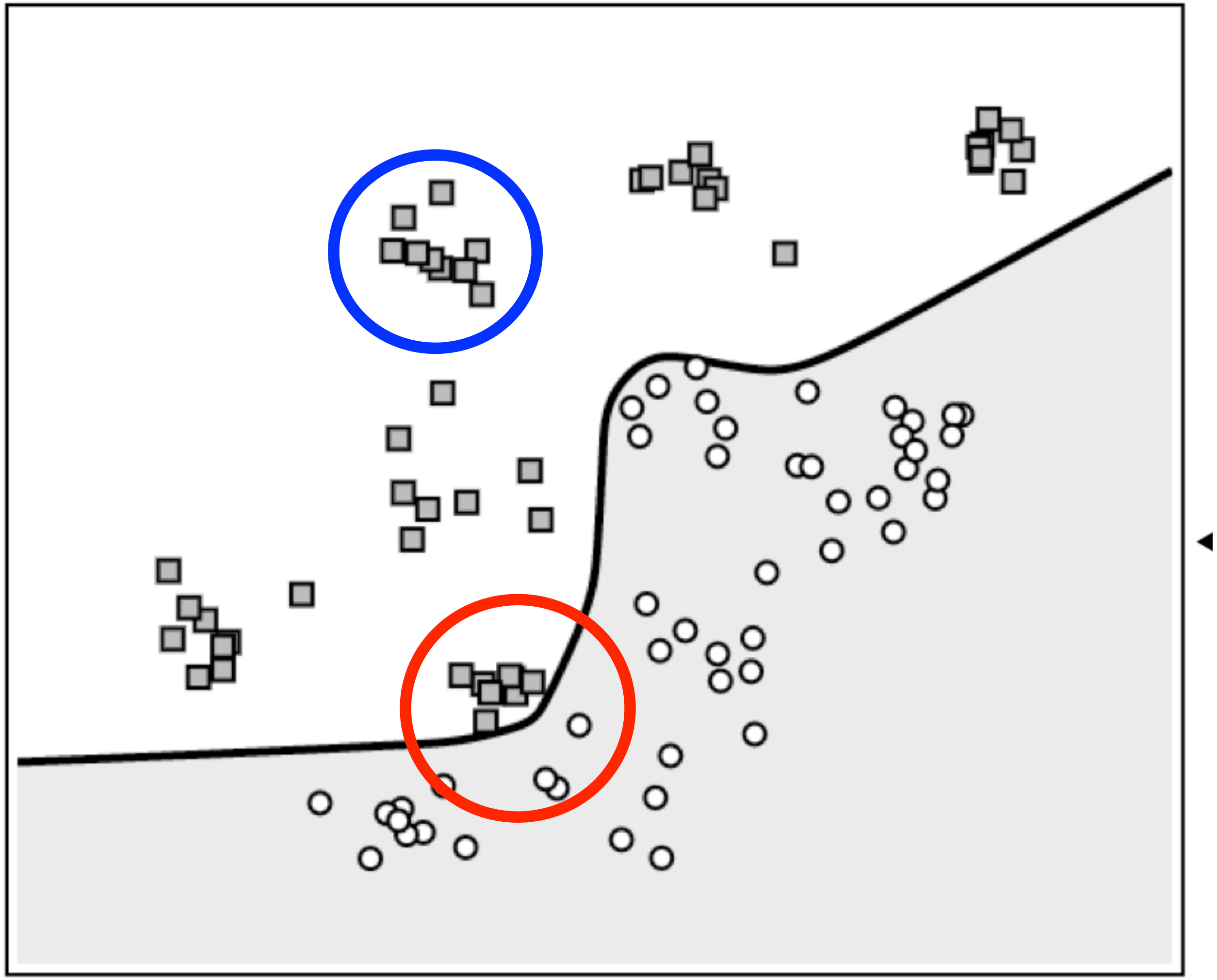


A New Bayesian Principle

"Data relevance" for free

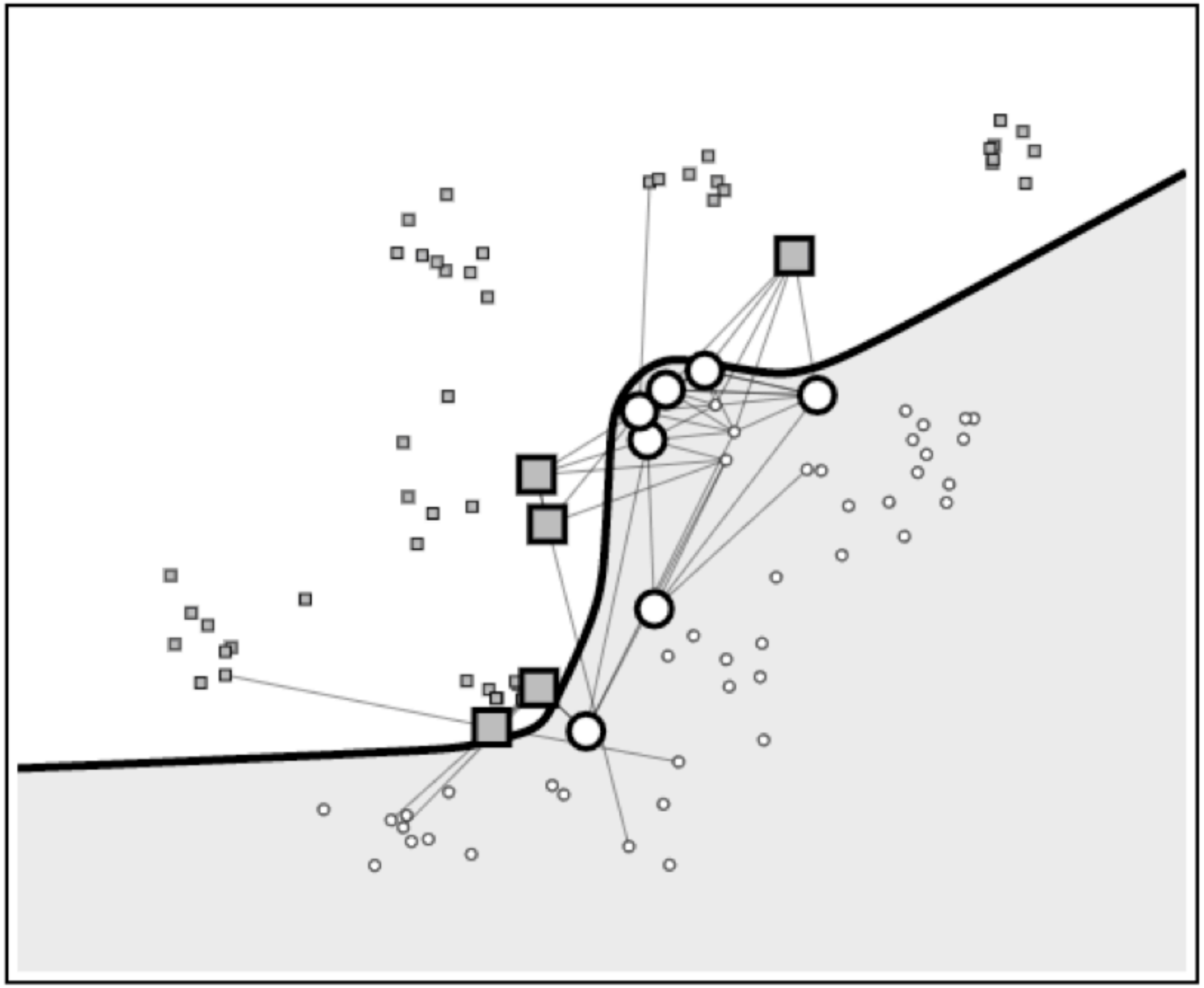
Defining Relevance

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



Model view vs Data view

Bayesian principles “automatically” define data-relevance.



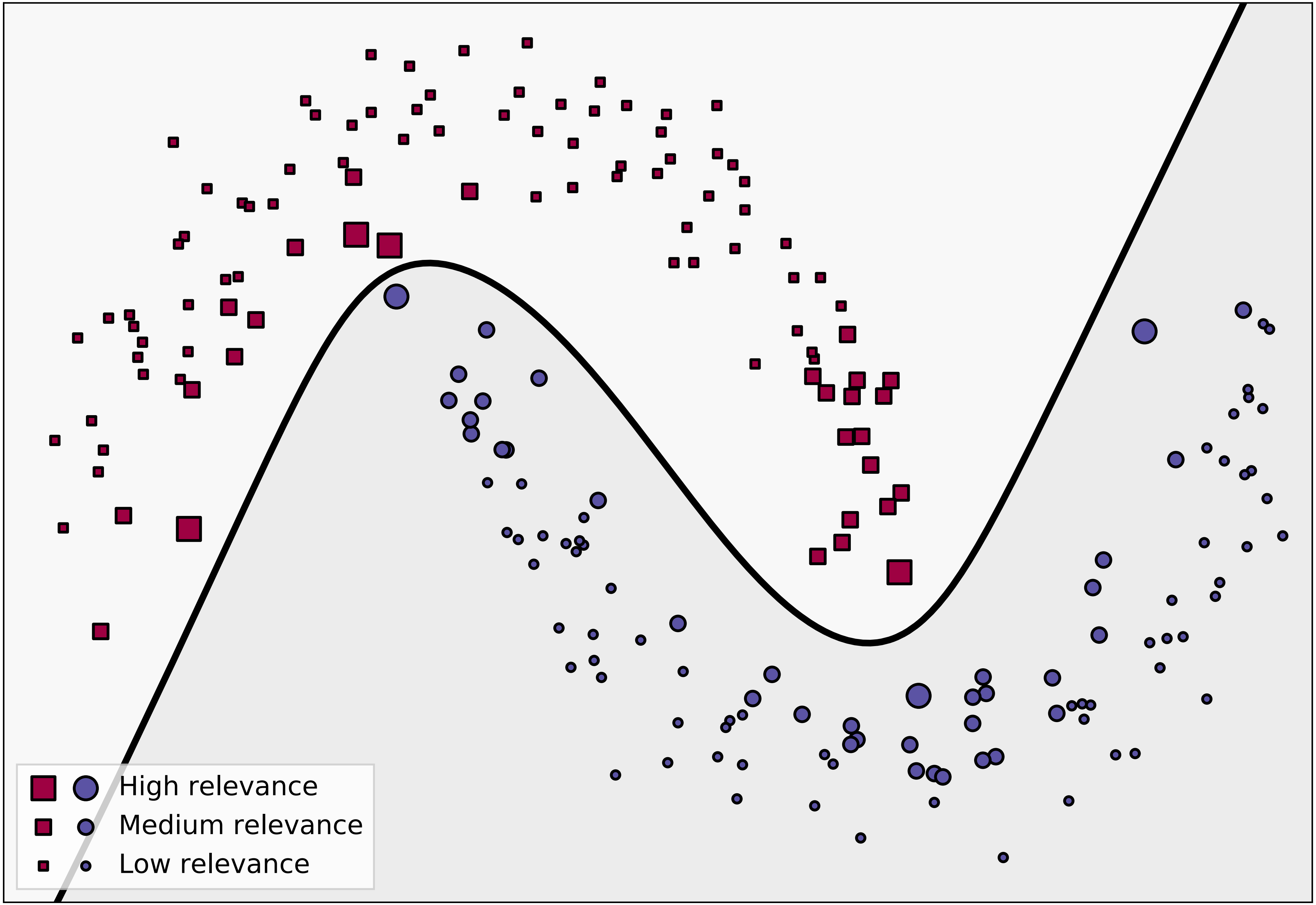
Data
view

Approximate Inference Turns Deep Networks into Gaussian Processes

Mohammad Emtiyaz Khan, Alexander Immer, Ehsan Abedi, Maciej Korzepa

(Submitted on 5 Jun 2019)

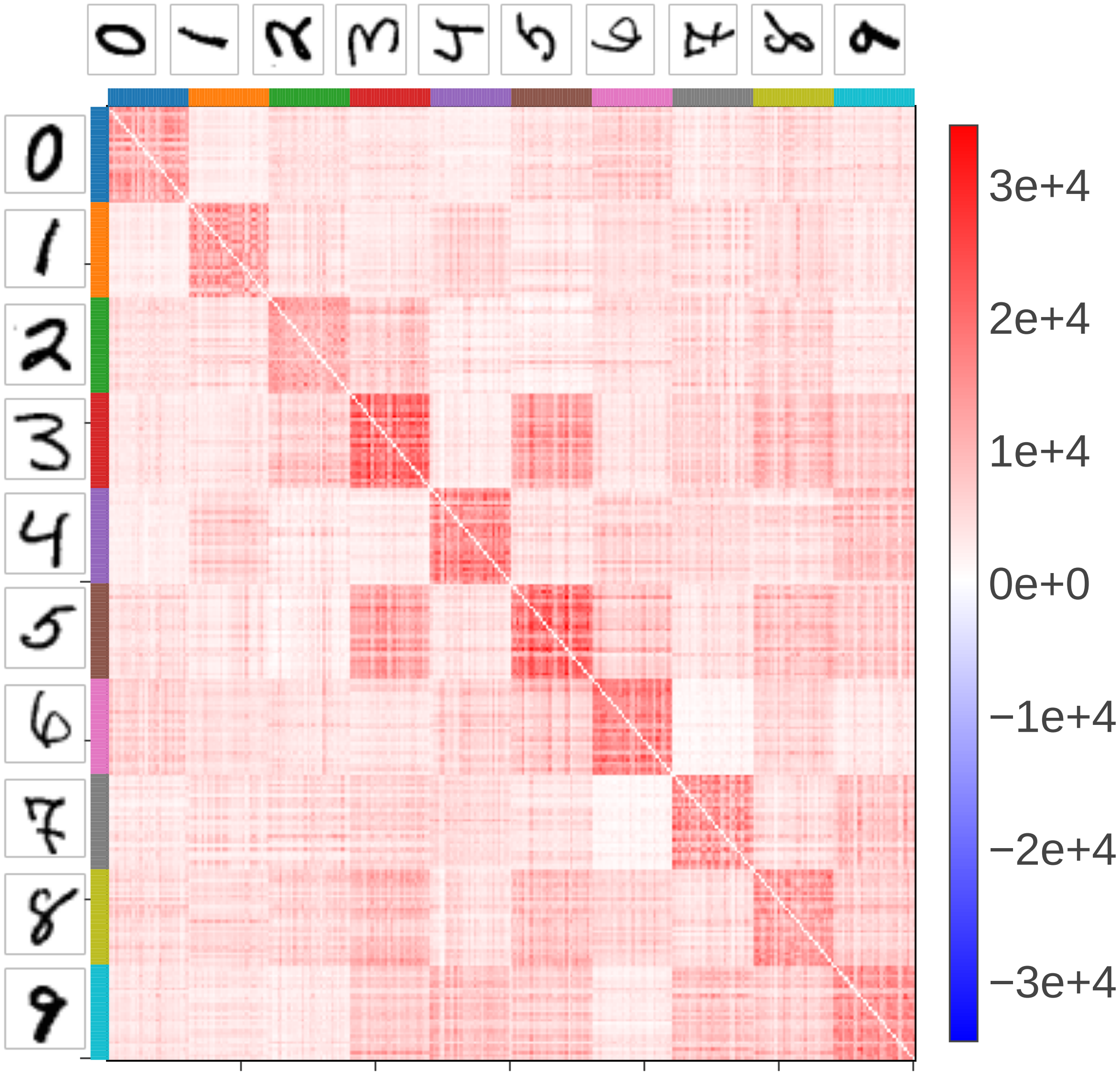
Deep neural networks (DNN) and Gaussian processes (GP) are two powerful models with several theoretical connections relating them, but the relationship between their training methods is not well understood. In this paper, we show that certain Gaussian posterior approximations for Bayesian DNNs are equivalent to GP posteriors. As a result, we can obtain a GP kernel and a nonlinear feature map simply by training the DNN. Surprisingly, the resulting kernel is the neural tangent kernel which has desirable theoretical properties for infinitely-wide DNNs. We show feature maps obtained on real datasets and demonstrate the use of the GP marginal likelihood to tune hyperparameters of DNNs. Our work aims to facilitate further research on combining DNNs and GPs in practical settings.


High relevance
Medium relevance
Low relevance

Random

Relevant

Similarity (Kernel) Matrix



Bayesian Duality Principle (WIP)

NeurIPS 2019

likelihood prior

$$\sum_{i=1}^{N} \ell(y_i, f_\theta(x_i)) + \gamma \theta^\top \theta$$

neural network

$$\approx \sum_{i=1}^{N} g_i^\top \theta - \theta^\top H_i \theta + \gamma \theta^\top \theta$$

$$\approx \sum_{i=1}^{N} \frac{1}{\sigma_i^2} [\tilde{y}_i - \phi_i(x_i)^\top \theta]^2 + \gamma \theta^\top \theta$$

$$q(\theta) := \mathcal{N}(\theta | m, \mathrm{Diag}(v))$$

$$\theta \leftarrow \mu + \epsilon, \;\; \text{where } \epsilon \sim \mathcal{N}(0, Ns + \lambda)$$

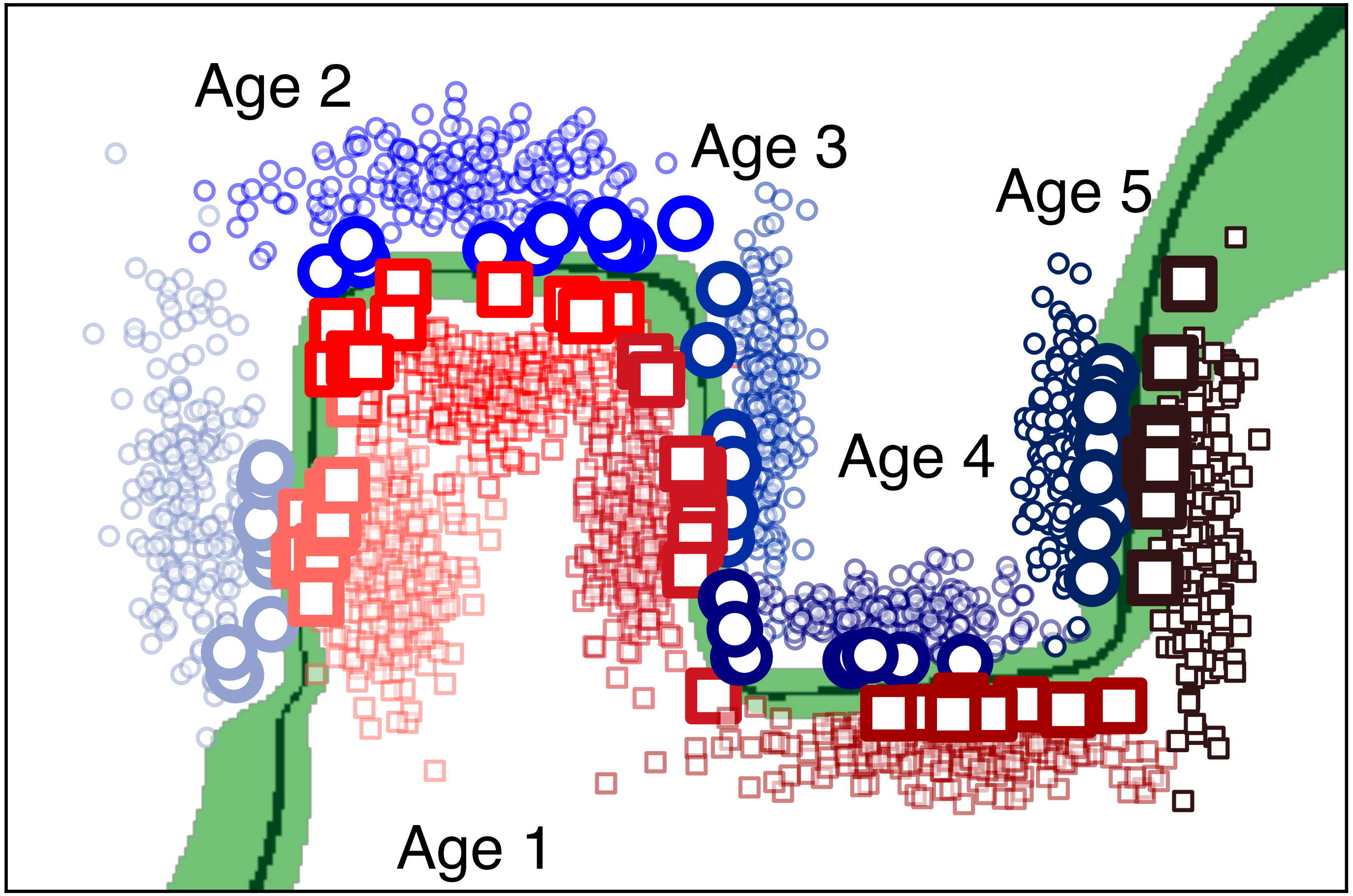
$$g \leftarrow \frac{1}{M} \sum_i \nabla_\theta \ell(y_i, f_\theta(x_i))$$

$$s \leftarrow (1-\beta)s + \beta \frac{1}{M} \sum_i [\nabla_\theta \ell(y_i, f_\theta(x_i))]^2$$

$$\mu \leftarrow \mu + \alpha \, \frac{g + \lambda\mu/N}{s + \lambda/N}$$

Deep Continual Learning with Bayes

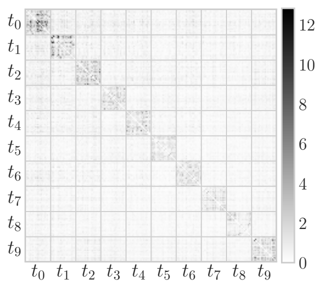
FROMP: Functional Regularization of Memorable Past
"Identify, Memorize, and Regularize"



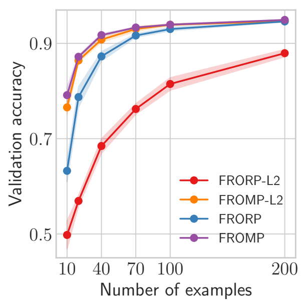
Works Well on Standard Benchmarks

On MNIST and CIFAR-100, we get state-of-the-art results!

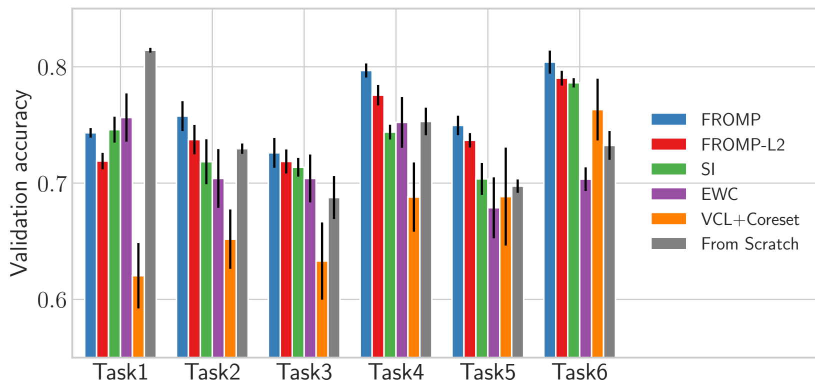
Method	Permuted MNIST	Split MNIST
DLP (Smola et al., 2003)	82%	61.2%
EWC (Kirkpatrick et al., 2017)	84%	63.1%
SI (Zenke et al., 2017)	86%	98.9%
Improved VCL (Swaroop et al., 2019)	93% ± 1	98.4% ± 0.4
+ random Coreset	**94.6**% ± 0.3 (200 p/t)	98.2% ± 0.4 (40 p/t)
FRCL-RND (Titsias et al., 2019)	94.2% ± 0.1 (200 p/t)	96.7% ± 1.0 (40 p/t)
FRCL-TR (Titsias et al., 2019)	94.3% ± 0.1 (200 p/t)	97.4% ± 0.6 (40 p/t)
FRORP-L_2	87.9% ± 0.7 (200 p/t)	98.5% ± 0.2 (40 p/t)
FROMP-L_2	94.6% ± 0.1 (200 p/t)	98.7% ± 0.1 (40 p/t)
FRORP	94.6% ± 0.1 (200 p/t)	**99.0**% ± 0.1 (40 p/t)
FROMP	**94.9**% ± 0.1 (200 p/t)	**99.0**% ± 0.1 (40 p/t)



(b) *memorable past* kernel

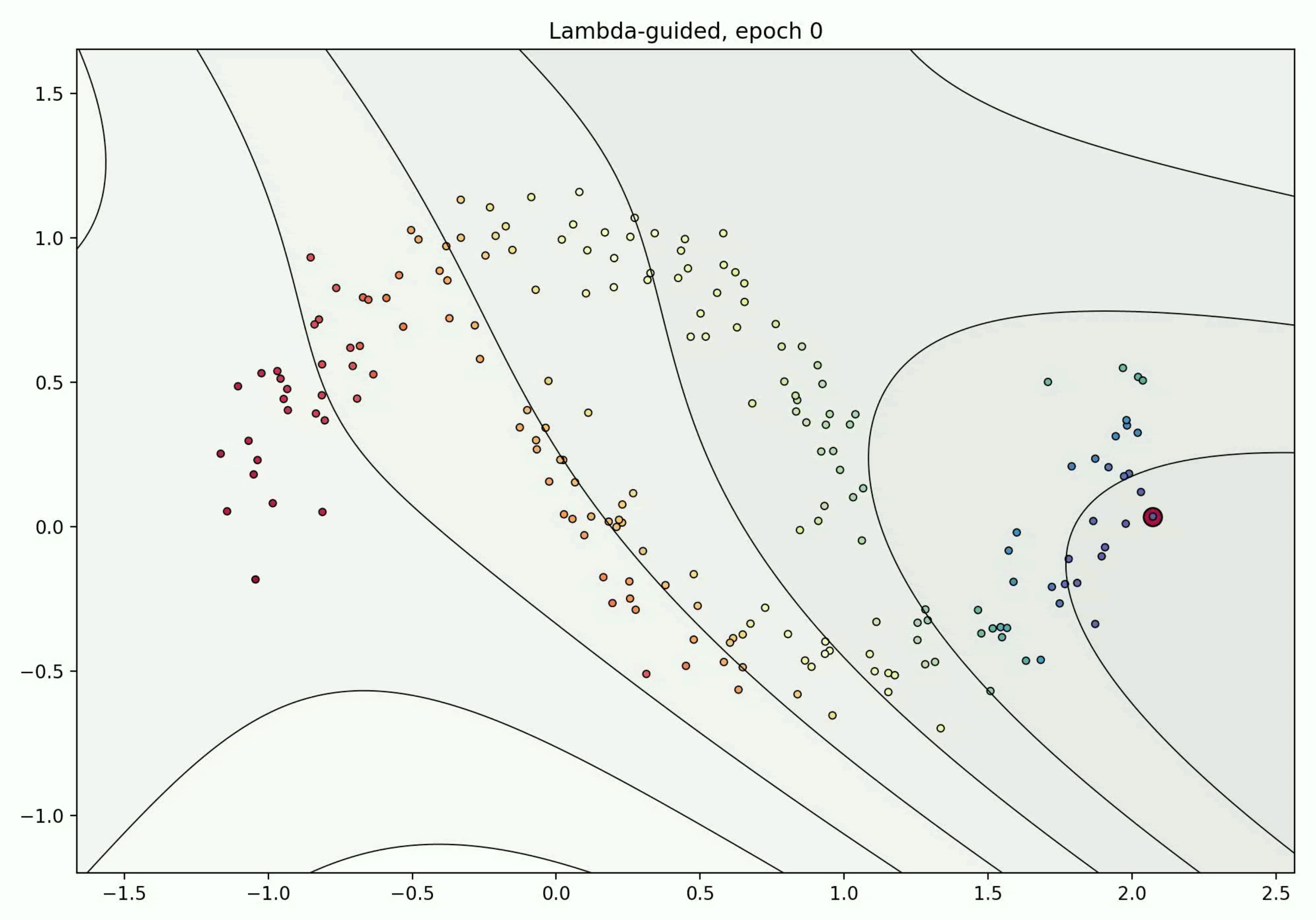


(c) accuracy vs. memory size



Relevance of Examples

Given a minibatch at each iteration, we select examples with less noise (low variance of epsilon_i in the approximated linear model).



(By Roman Bachmann)

Learning-Algorithms from Bayesian Principles

- Practical Bayesian principles.
- Bayesian learning rule
  - a generalization of many learning-algorithms,
    - Classical (least-squares, Newton, HMM, Kalman.. etc).
    - Deep Learning (SGD, RMSprop, Adam).
- Data relevance
- Continual Learning with Bayes
- Impact: Everything with one common principle.

References

Available at https://emtiyaz.github.io/publications.html

Conjugate-Computation Variational Inference : Converting Variational Inference in Non-Conjugate Models to Inferences in Conjugate Models, (AIstats 2017) **M.E. Khan** and W. Lin [Paper] [Code for Logistic Reg

Fast and Scalable Bayesian Deep Learning by Weight-Perturbation in Adam, (ICML 2018) **M.E. Khan**, D. Nielsen, V. Tangkaratt, W. Lin, Y. Gal, and A. Srivastava, [ArXiv Version] [Code] [Slides]

Practical Deep Learning with Bayesian Principles, (Under review) K. Osawa, S. Swaroop, A. Jain, R. Eschenhagen, R.E. Turner, R. Yokota, **M.E. Khan**. [arXiv]

Approximate Inference Turns Deep Networks into Gaussian Processes, (Under review) **M.E. Khan**, A. Immer, E. Abedi, M. korzepa. [arXiv]

Learning-Algorithms from Bayesian Principles

A long paper to be released before my NeurIPS tutorial

Mon Dec 9th 08:30 -- 10:30 AM @ West Hall A Tutorial

Deep Learning with Bayesian Principles

Mohammad Emtiyaz Khan

Emtiyaz Khan »

Deep learning and Bayesian learning are considered two entirely different fields often used in complementary settings. It is clear that combining ideas from the two fields would be beneficial, but how can we achieve this given their fundamental differences?

This tutorial will introduce modern Bayesian principles to bridge this gap. Using these principles, we can derive a range of learning-algorithms as special cases, e.g., from classical algorithms, such as linear regression and forward-backward algorithms, to modern deep-learning algorithms, such as SGD, RMSprop and Adam. This view then enables new ways to improve aspects of deep learning, e.g., with uncertainty, robustness, and interpretation. It also enables the design of new methods to tackle challenging problems, such as those arising in active learning, continual learning, reinforcement learning, etc.

Overall, our goal is to bring Bayesians and deep-learners closer than ever before, and motivate them to work together to solve challenging real-world problems by combining their strengths.

A 5 page review

Fast yet Simple Natural-Gradient Descent for Variational Inference in Complex Models

Mohammad Emtiyaz Khan
RIKEN Center for Advanced Intelligence Project
Tokyo, Japan
emtiyaz.khan@riken.jp

Didrik Nielsen
RIKEN Center for Advanced Intelligence Project
Tokyo, Japan
didrik.nielsen@riken.jp

***Abstract*—Bayesian inference plays an important role in advancing machine learning, but faces computational challenges when applied to complex models such as deep neural networks. Variational inference circumvents these challenges by formulating Bayesian inference as an optimization problem and solving it using gradient-based optimization. In this paper, we argue in favor of *natural-gradient* approaches which, unlike their *gradient-based* counterparts, can improve convergence by exploiting the information geometry of the solutions. We show how to derive fast yet simple natural-gradient updates by using a duality associated with exponential-family distributions. An attractive feature of these methods is that, by using natural-gradients, they are able to extract accurate local approximations for individual model components. We summarize recent results for Bayesian deep learning showing the superiority of natural-gradient approaches over their gradient counterparts.**

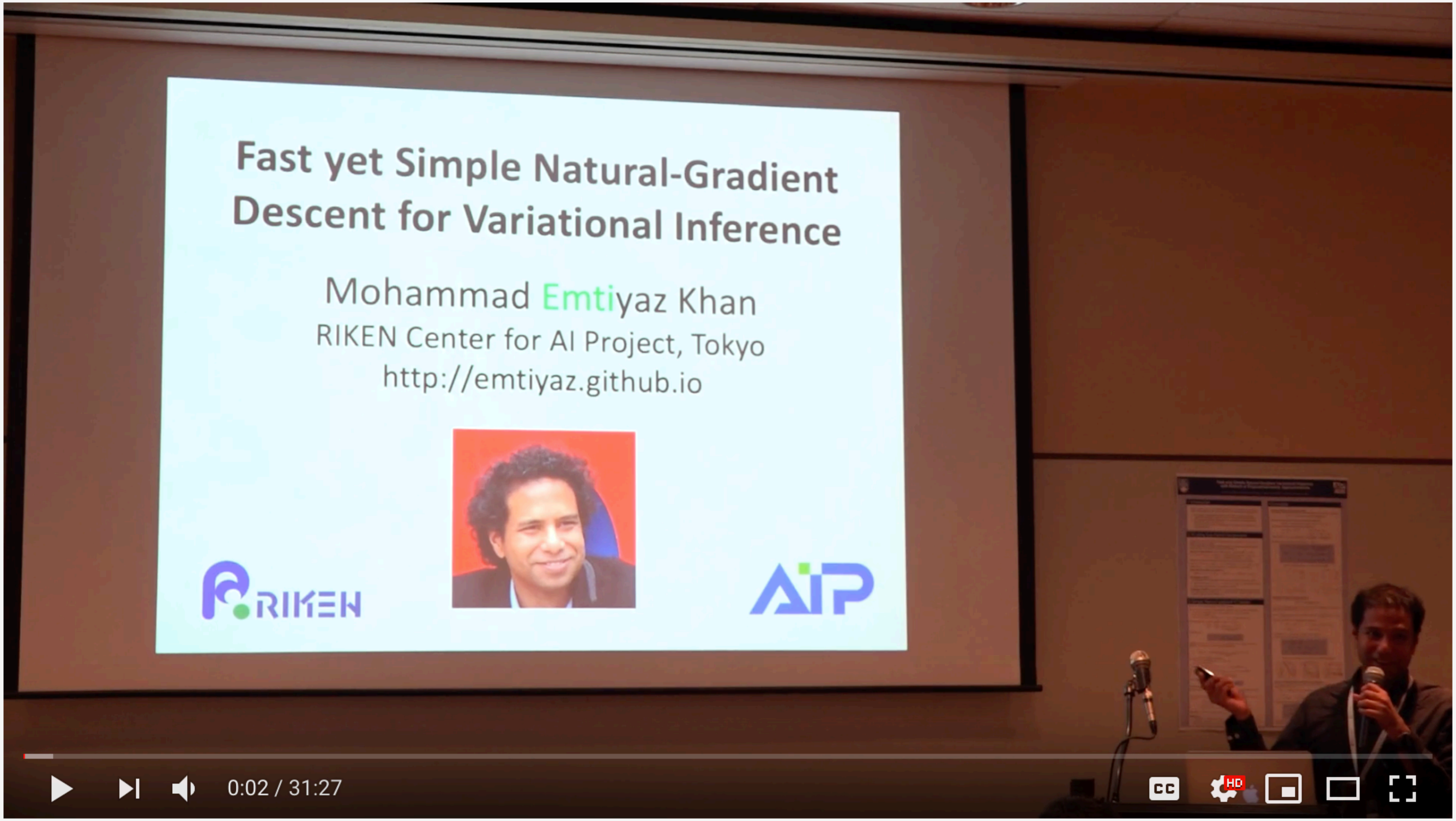
***Index Terms*—Bayesian inference, variational inference, natural gradients, stochastic gradients, information geometry, exponential-family distributions, nonconjugate models.**

prove the rate of convergence [7]–[9]. Unfortunately, these approaches only apply to a restricted class of models known as *conditionally-conjugate* models, and do not work for non-conjugate models such as Bayesian neural networks.

This paper discusses some recent methods that generalize the use of natural gradients to such large and complex non-conjugate models. We show that, for exponential-family approximations, a duality between their natural and expectation parameter-spaces enables a simple natural-gradient update. The resulting updates are equivalent to a recently proposed method called Conjugate-computation Variational Inference (CVI) [10]. An attractive feature of the method is that it naturally obtains *local* exponential-family approximations for individual model components. We discuss the application of the CVI method to Bayesian neural networks and show some recent results from a recent work [11] demonstrating

YouTube DE

Search



Emtiyaz Khan: Fast yet Simple Natural-Gradient Descent for Variational Inference

Acknowledgements

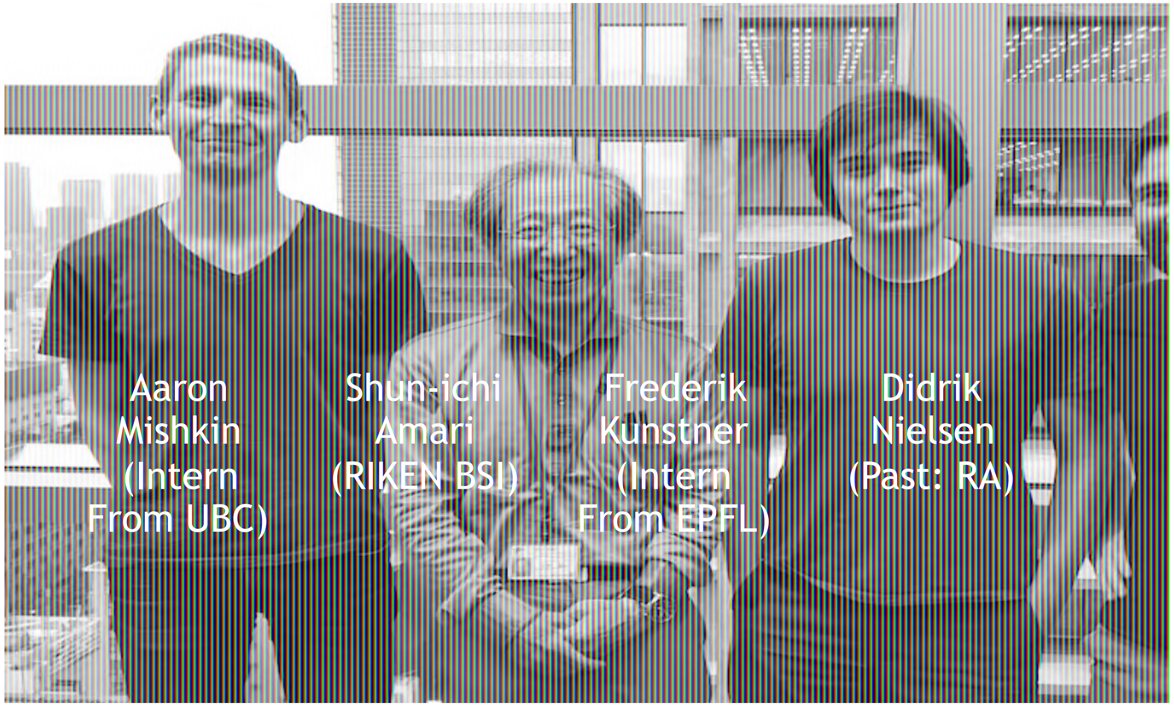
Slides, papers, & code are at emtiyaz.github.io

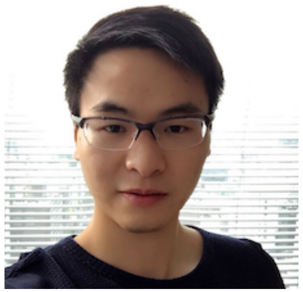
Wu Lin
(Past: RA)

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)

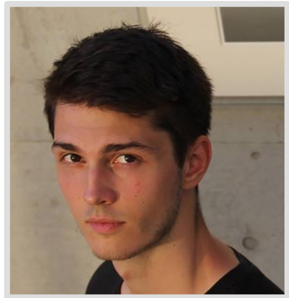
Acknowledgements

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

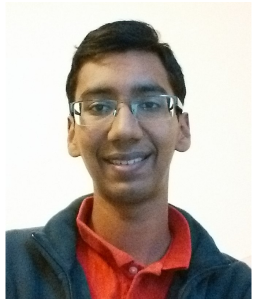
Kazuki Osawa
(Tokyo Tech)

Rio Yokota
(Tokyo Tech)

Anirudh Jain
(Intern from
IIT-ISM, India)



Runa Eschenhagen
(Intern from
University of
Osnabruck)

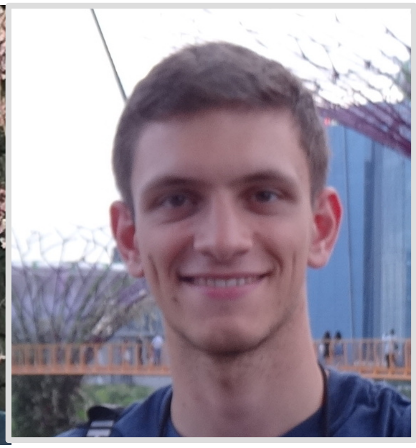


Siddharth
Swaroop
(University of
Cambridge)

Rich Turner
(University of
Cambridge)

Alexander Immer
(Intern from EPFL)

Ehsan Abedi
(Intern from EPFL)



Maciej Korzepa
(Intern from TU Denmark)

Pierre Alquier
(RIKEN AIP)

Havard Rue
(KAUST)

Approximate Bayesian Inference Team

Looking for interns, research assistants, post-docs, and collaborators