Karlsruhe Institute of Technology

Uncertainty in Deep Learning: A Probabilistic Robotics Perspective

Jongseok Lee

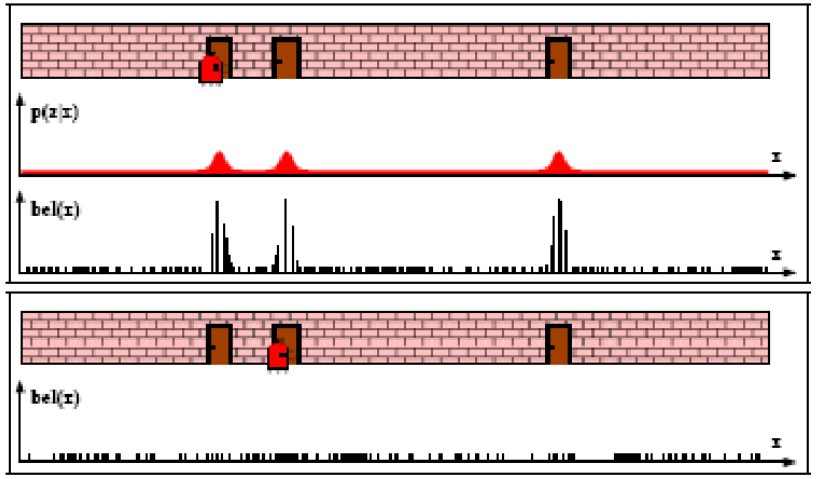
PhD Defense

Is this our Sisyphean climb?

Karlsruhe Institute of Technology

Probabilistic Robotics

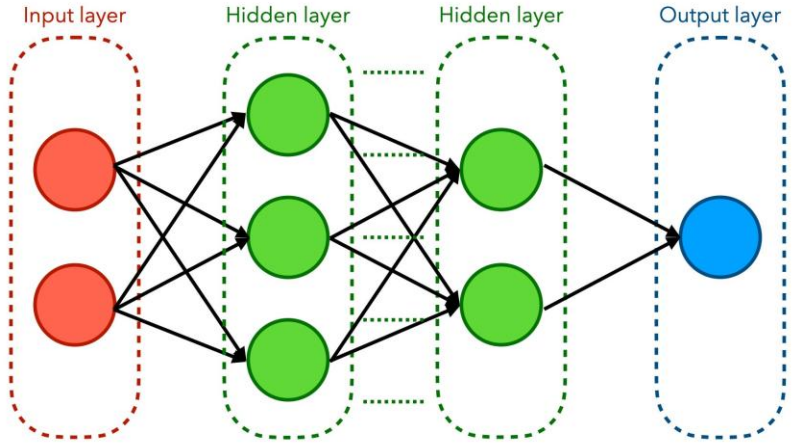
Since 1990s



- Do not optimize but average.
- Pay tribute to uncertainty.

AI-based Robotics

Since 2010s



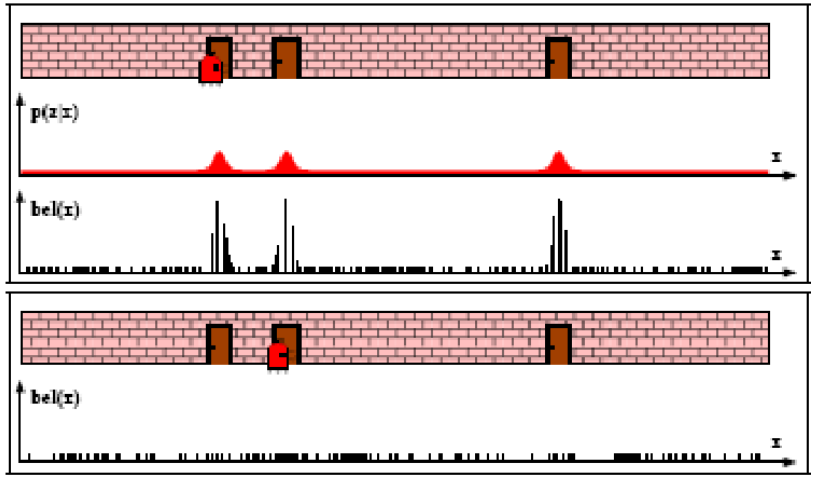
- Optimize as much as possible.
- Worships performance, not doubt.

No Sisyphus – Hercule‘s bridge

Karlsruhe Institute of Technology

Probabilistic Robotics

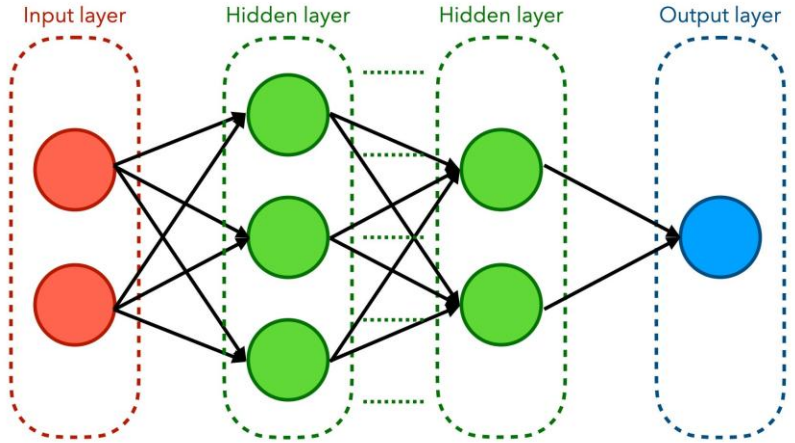
Since 1990s



- Do not optimize but average.
- Pay tribute to uncertainty.

AI-based Robotics

Since 2010s



- Optimize as much as possible.
- Worships performance, not doubt.

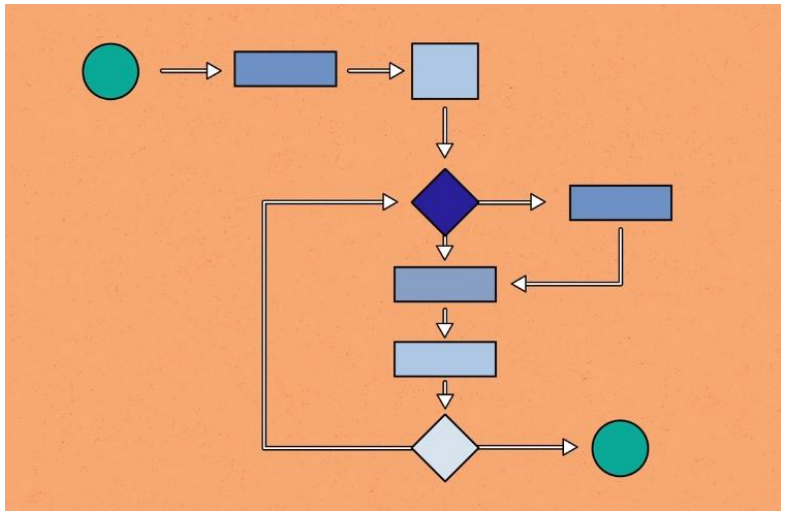
AI-based robotics that reason about uncertainty – no maximum likelihood

Why should we care?

Safety & Risk

Trustworthy AI

Better Algorithms



Preamble: Bayesian statistics

Recap 1: Probability is defined as degree of one‘s belief about an event.

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

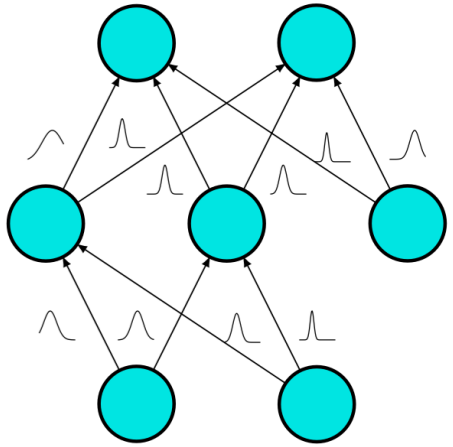
posterior ↓ prior ↓

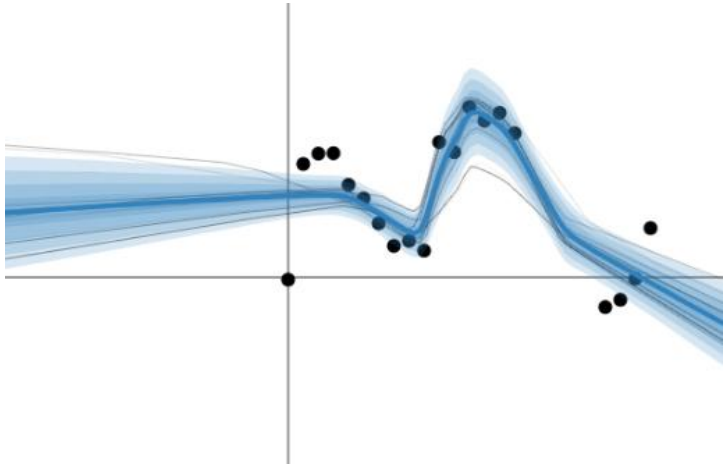
Recap 2: Bayes Theorem updates prior belief to posterior given data.

Preamble: Bayesian Deep Learning

Intractable!

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





Intractable!

$$p(y^*|x^*, D) = \int p(y^*|x^*, \theta)p(\theta|D)d\theta$$

- \+ Inclusion of domain knowledge via prior.
- \+ Quantifies uncertainty about the model.
- \+ Average different hypothesis about events.

Note: θ is neural network (parameters) & D is training data.

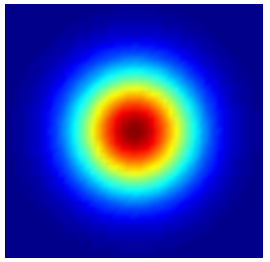
Preamble: Being Bayesian is hard

- Tisby et al. 1989
- Mackay 1992
- Hinton and Van Camp 1993
- Neal 1995

AI winter

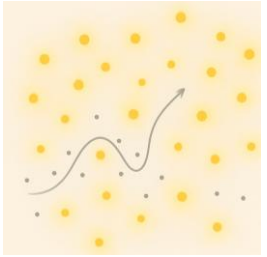
- Graves 2011
- Hernandez-Lobato and Adams 2015
- **MC-dropout** (Gal and Ghahramani 2016)
- Deep Ensemble (Lakshminarayanan et al. 2017)

When I started



Uninfomative prior

Bernoulli posterior



Sampling

- Prior specification is non-trivial.
- Posterior is crudely approximated.
- Sampling is computationally expensive.

Key research objective

Challenges:

- Prior specification is non-trivial.
- Posterior is crudely approximated.
- Sampling is more expensive.

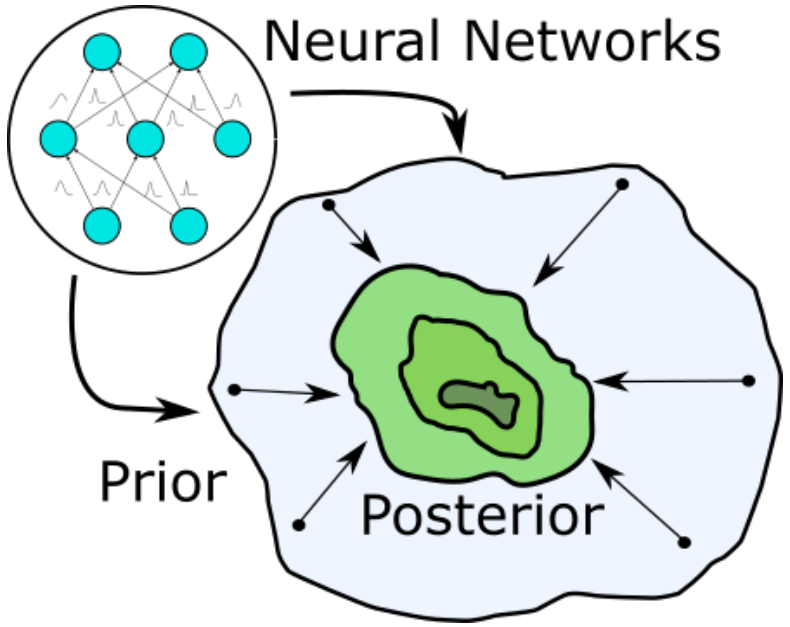
Promises:

- Inclusion of domain knowledge via prior.
- Quantifies uncertainty about the model.
- Average different hypothesis about events.

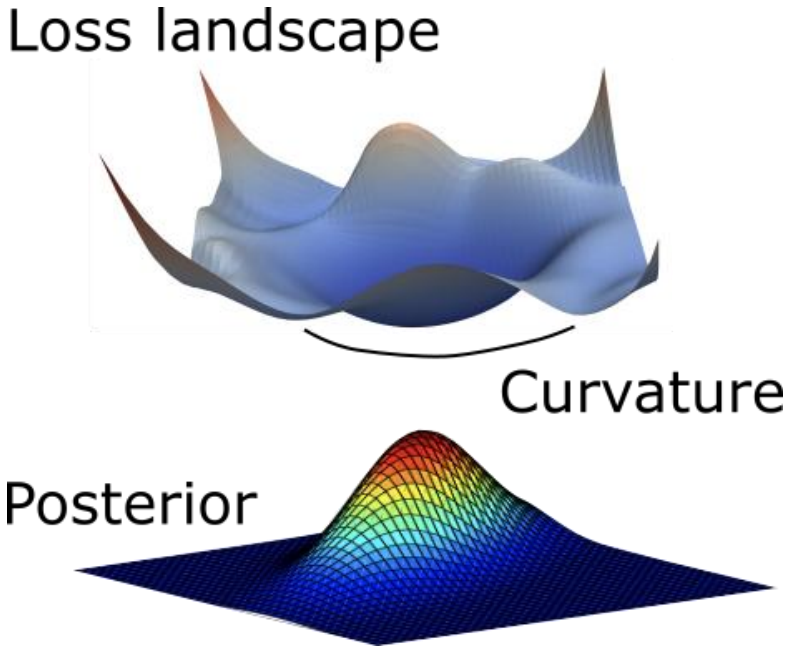
Bayesian methods that are well applicable in AI-based robotics.

Contributions

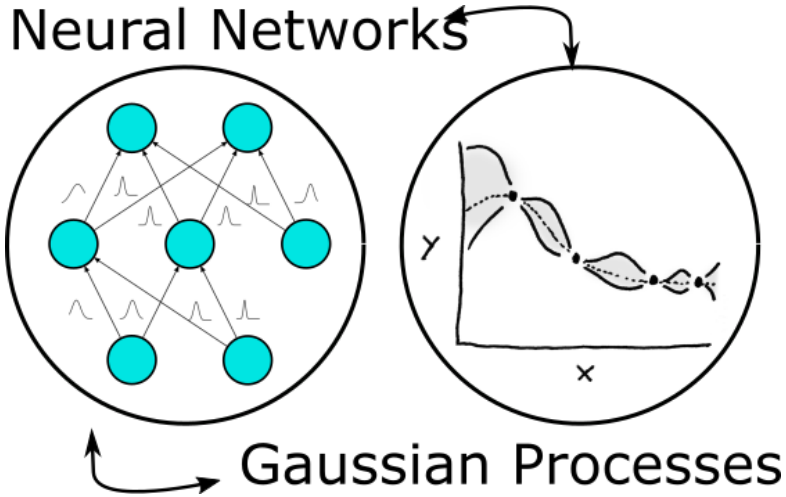
On Priors



On Posteriors

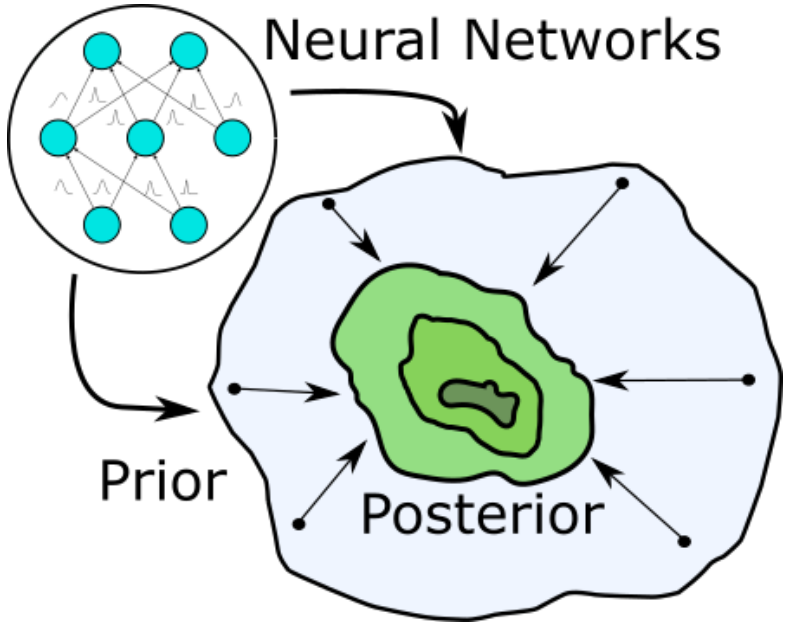


On Predictions



Overview

On Priors

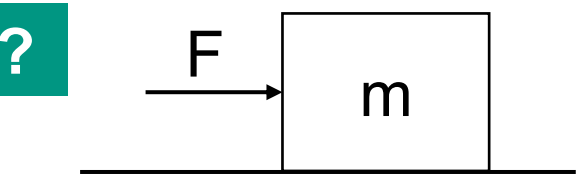


How to specify more meaningful priors for uncertainty and generalization?

Challenge: Issue of interpretability

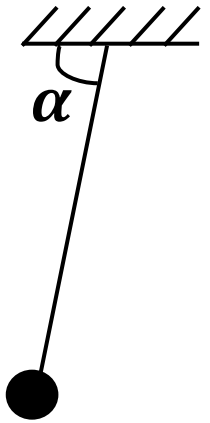
$F=ma$ where a constant m is a random variable. Prior $p(m)$?

- Gamma distribution. Mass m is a quantity of positive values.



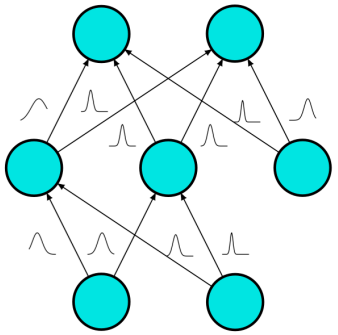
A pendulum where an angle α is a random variable. Prior $p(\alpha)$?

- Von Mises distribution to reflect that angle $\boldsymbol{\alpha}$ ranges from zero to 2π.



A network with the weights θ as random variable. Prior $p(\theta)$?

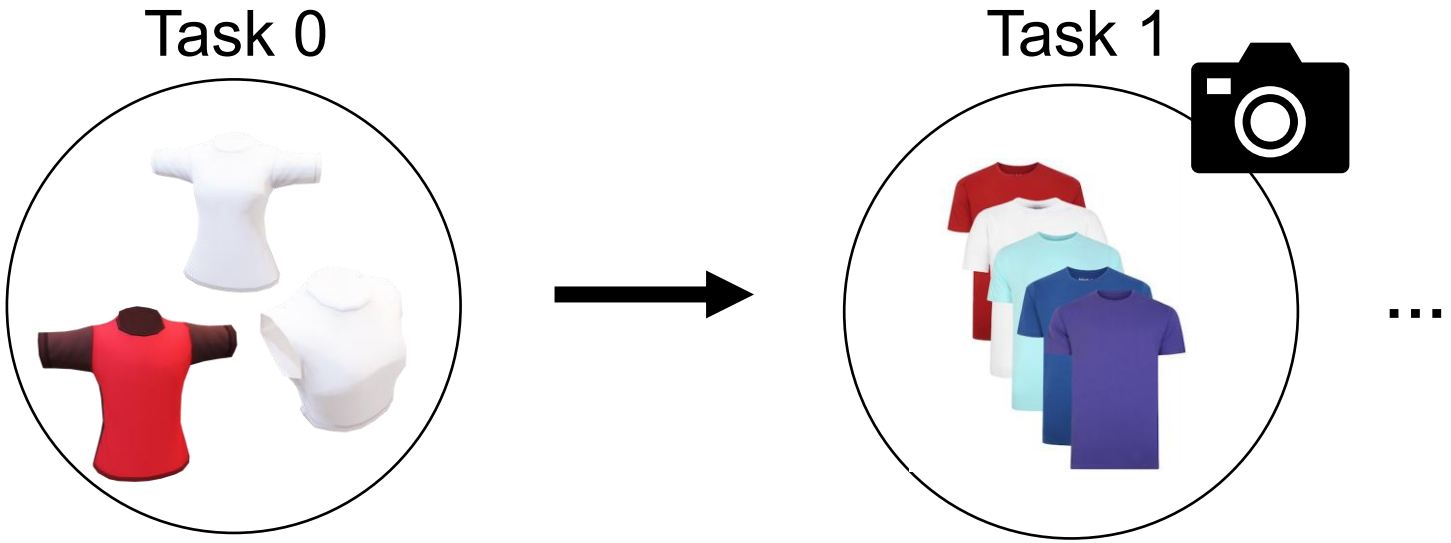
- Zero mean isotropic Gaussian: $\boldsymbol{p(\theta) = \mathcal{N}(0, \gamma I)}$.
- Despite reported signs of prior misspecification (Fortuin, 2022).



Schnaus, D*., Lee, J. *, Cremers, D., & Triebel, R. (2023). Learning expressive priors for generalization and uncertainty estimation in neural networks. In *International Conference on Machine Learning* (pp. 30252-30284). *equal contributions

Idea: Sequential Bayes for neural networks

- Learning priors from data:

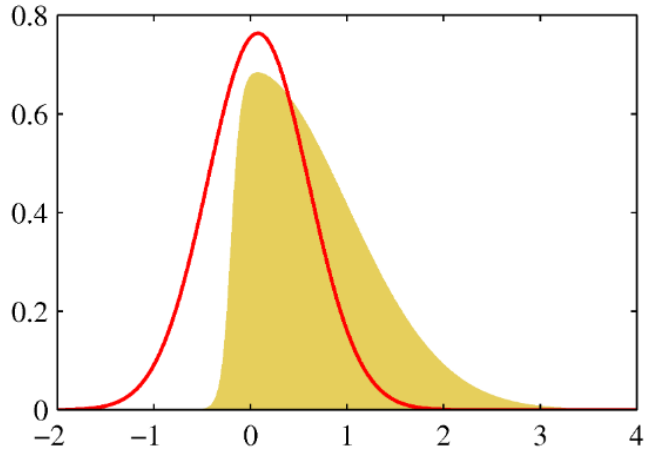


0. $p(\theta^{(0)}) = \mathcal{N}(\theta^{(0)}|0, \tau I)$

1. $p(\theta^{(0)}|D^{(0)})$

2. $p(\theta^{(1)}) = p(\theta^{(0)}|D^{(0)})$

3. $p(\theta^{(1)}|D^{(1)})$

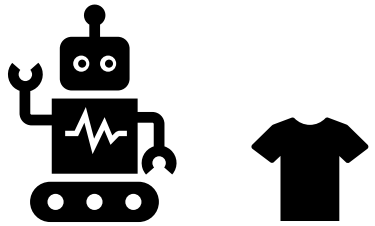


$$p(\theta|D) \approx \mathcal{N}(\theta|\hat{\theta}, H^{-1})$$

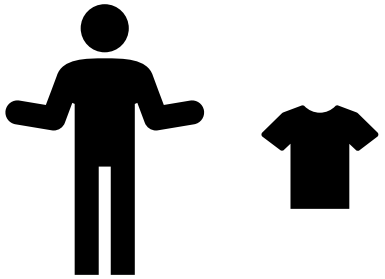
- $\hat{\theta} \in \operatorname{argmax} p(\theta|D)$
- $H = H_{\text{likelihood}} + H_{\text{prior}}$

Schnaus, D*., Lee, J. *, Cremers, D., & Triebel, R. (2023). Learning expressive priors for generalization and uncertainty estimation in neural networks. In *International Conference on Machine Learning* (pp. 30252-30284). *equal contributions

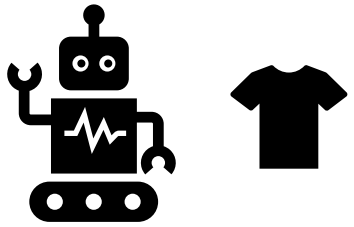
Evaluation: Robot asks help and learn



I don‘t know what this object is.
Can you help?



This is a T-shirt.
Let me demonstrate to you.



Let me train.
Now I know about this object.

- **Task:** Uncertainty estimation.

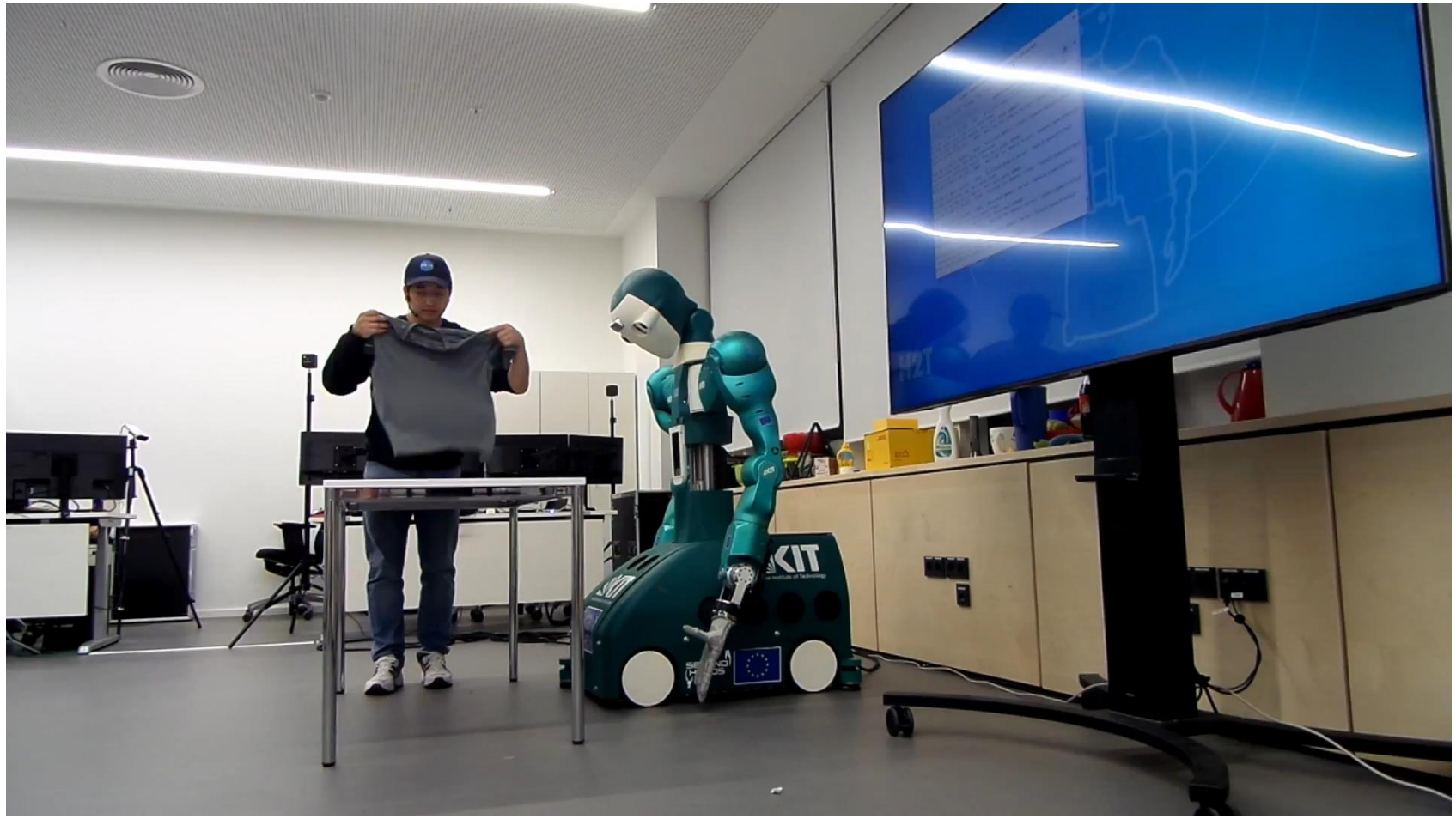
→ Evaluates prior specification.

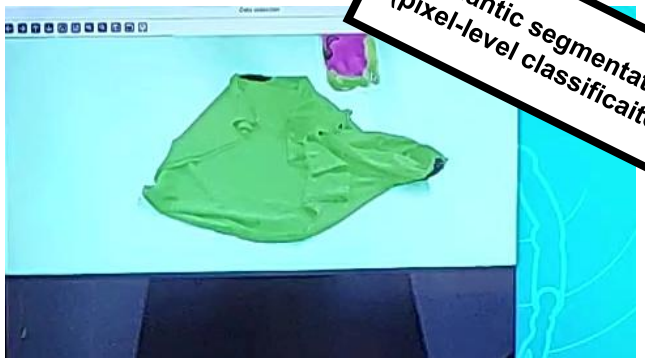
- **Task:** Learning from small data.

→ Evaluates generalization.

Lee, J., Birr, T., Triebel, R., & Asfour, T. (2025). CLEVER: Stream-Based Active Learning for Robust Semantic Perception From Human Instructions. *IEEE Robotics and Automation Letters*.

Demonstration on a humanoid robot



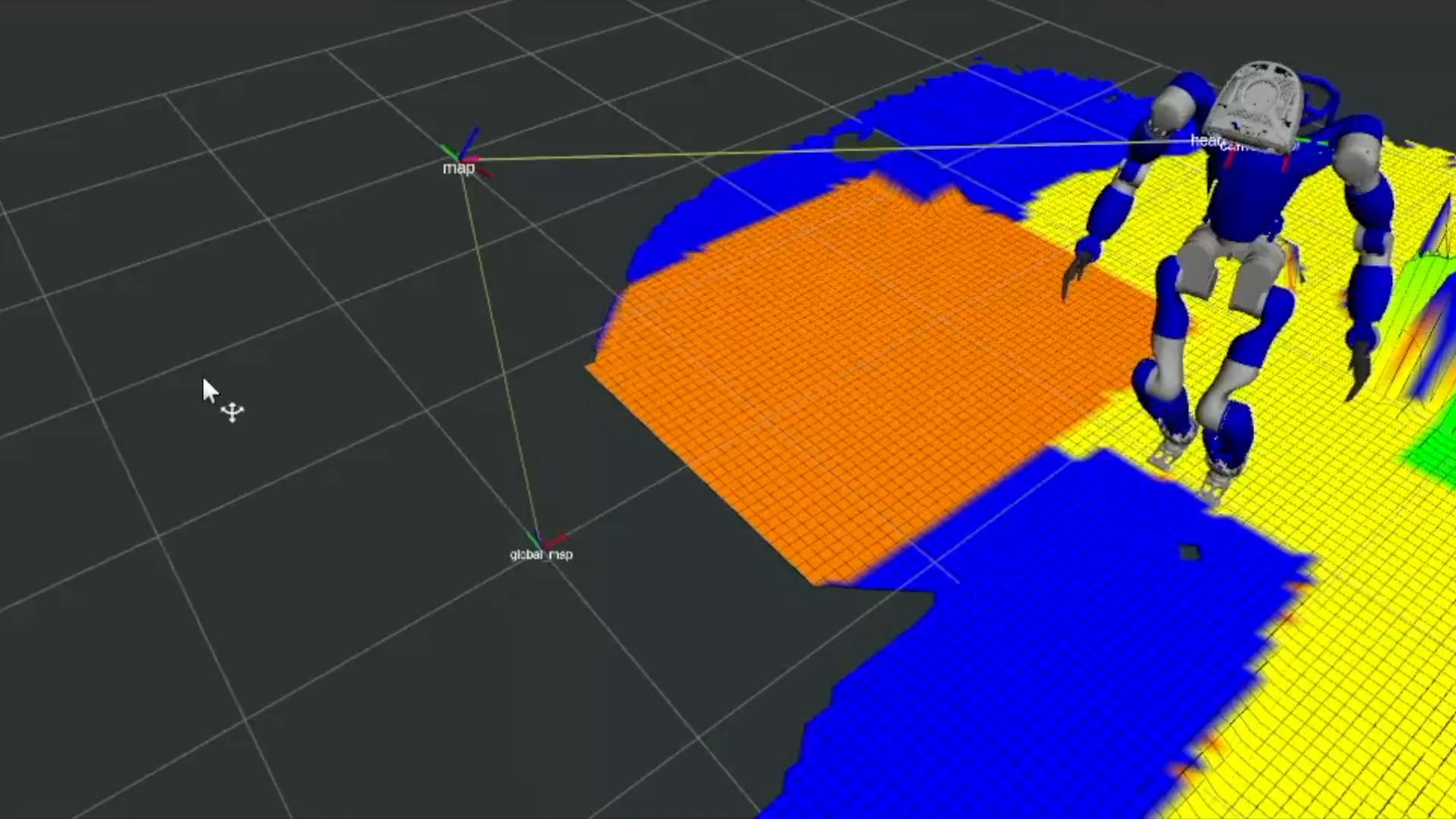


	Success rate	ECE
CLEVERv1	**0.887±0.049**	**0.060±0.017**
CLEVERv2	0.826±0.053	0.092±0.036
Vanilla	0.801±0.054	0.177±0.021

	Precision	Nr. Queries
CLEVERv1	**0.933±0.020**	**1.539±0.544**
CLEVERv2	0.900±0.034	2.440±0.866
Vanilla	0.817±0.039	4.320±0.992

Prior helps uncertainty & generalization.

Lee, J., Birr, T., Triebel, R., & Asfour, T. (2025). CLEVER: Stream-Based Active Learning for Robust Semantic Perception From Human Instructions. *IEEE Robotics and Automation Letters*.

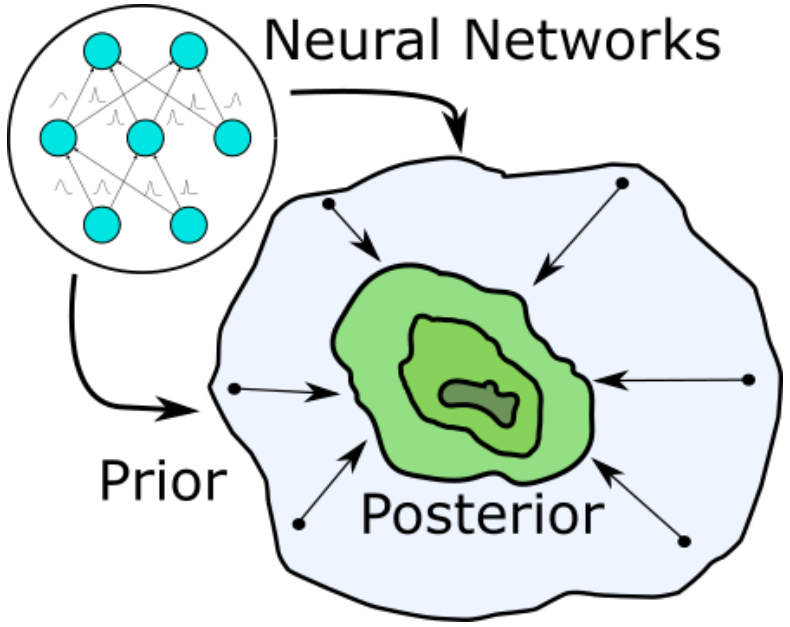

map
global_map


toro
DLR

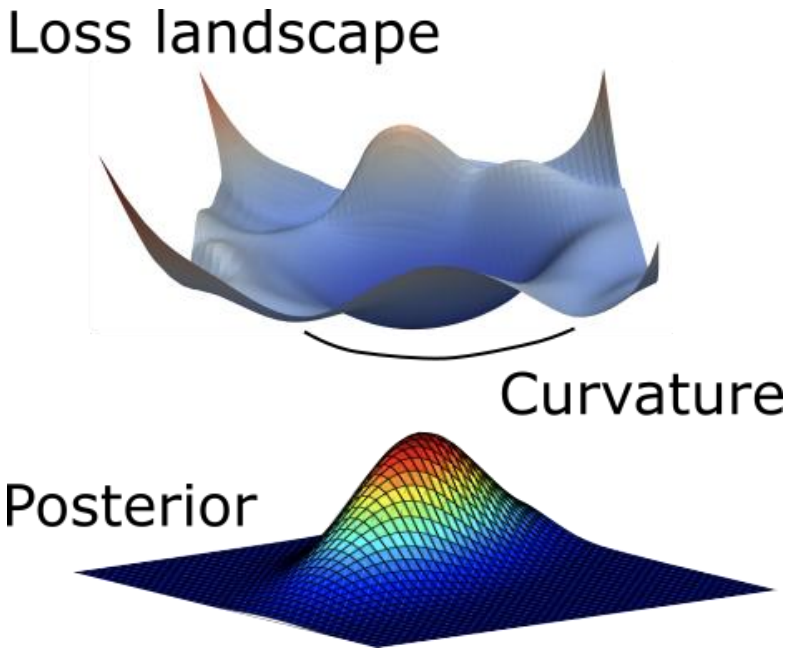

DLR
RIG STAGE
Deutsches Zentrum für Luft- und Raumfahrt
German Aerospace Center
Institute of Robotics and Mechatronics
Combining cutting-edge Research with present-day Business Challenges
TORO

Overview

On Priors



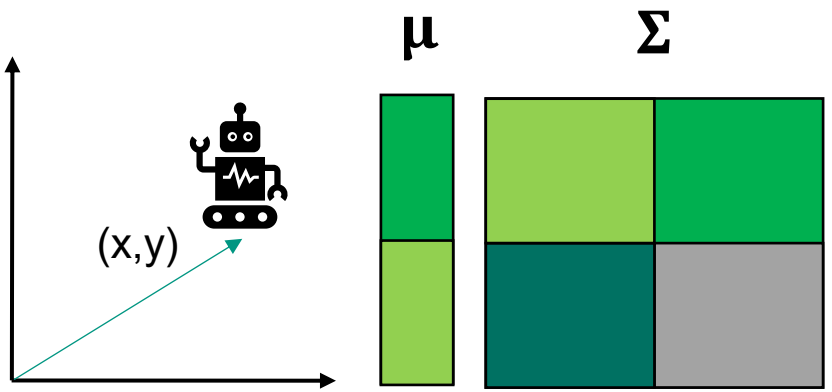
On Posteriors



How to infer complex posteriors while scaling to large dataset and models?

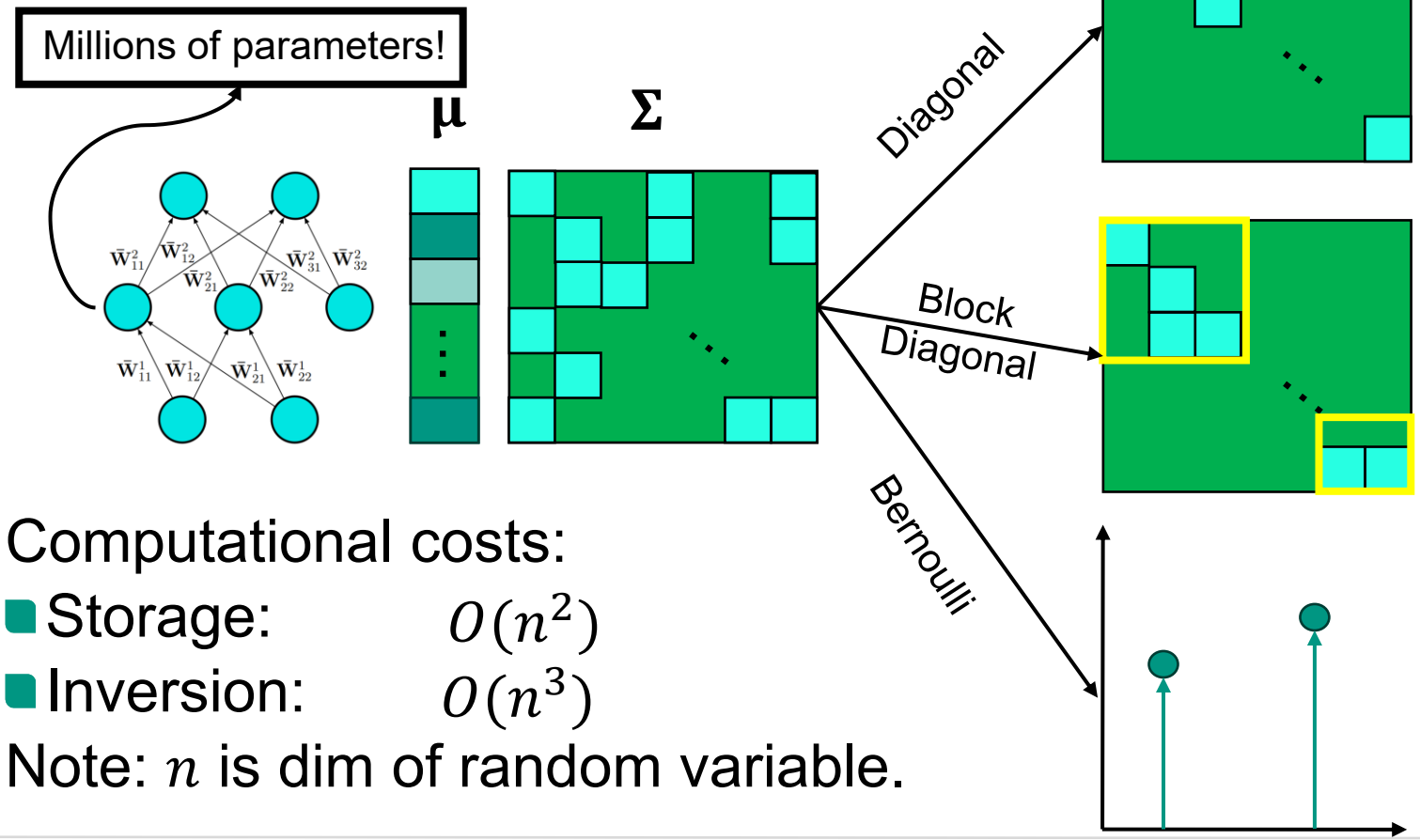
Challenge: Curse of Dimensionality

Simple 2D problem



$$\theta \propto e^{-\frac{1}{2}(\theta-\mu)^T \Sigma^{-1} (\theta-\mu)}$$

Deep Learning



Computational costs:

- Storage: $O(n^2)$
- Inversion: $O(n^3)$

Note: n is dim of random variable.

Lee, J., Humt, M., Feng, J., & Triebel, R (2020). Estimating model uncertainty of neural networks in sparse information form. In *International Conference on Machine Learning* (pp. 5702-5713).

Idea: Information Form

- Estimate uncertainty in information form:

  $p(\theta|D) \approx \mathcal{N}(\mu, \Sigma) = \mathcal{N}^{-1}(\eta, \Lambda)$ **with** $\eta = \mu^T \Sigma$ **and** $\Lambda = \Sigma^{-1}$.

- Why information form?

$$p(\alpha, \beta) = \mathcal{N}\left(\begin{bmatrix}\mu_\alpha \\ \mu_\beta\end{bmatrix}, \begin{bmatrix}\Sigma_{\alpha\alpha} & \Sigma_{\alpha\beta} \\ \Sigma_{\beta\alpha} & \Sigma_{\beta\beta}\end{bmatrix}\right) = \mathcal{N}^{-1}\left(\begin{bmatrix}\eta_\alpha \\ \eta_\beta\end{bmatrix}, \begin{bmatrix}\Lambda_{\alpha\alpha} & \Lambda_{\alpha\beta} \\ \Lambda_{\beta\alpha} & \Lambda_{\beta\beta}\end{bmatrix}\right)$$

- Goal: compute conditioning or posterior calculation $p(\alpha|\beta) = \frac{p(\alpha,\beta)}{p(\beta)}$:

  Covariance form: $\Sigma' = \Sigma_{\alpha\alpha} - \Sigma_{\alpha\beta}\Sigma_{\beta\beta}^{-1}\Sigma_{\beta\alpha}$.

  Information form: $\Lambda' = \Lambda_{\alpha\alpha}$.

Lee, J., Humt, M., Feng, J., & Triebel, R (2020). Estimating model uncertainty of neural networks in sparse information form. In *International Conference on Machine Learning* (pp. 5702-5713).

Idea: Information Form

$$O(n^3) \rightarrow O(L^3) \text{ for } L \ll n$$

- Estimate uncertainty in information form:

  $\boldsymbol{p(\theta|D) \approx \mathcal{N}(\mu, \Sigma) = \mathcal{N}^{-1}(\eta, \Lambda)}$ **with** $\boldsymbol{\eta = \mu^T \Sigma}$ **and** $\boldsymbol{\Lambda = \Sigma^{-1}}$.

- Why information form?

$$\boldsymbol{p(\alpha, \beta) = \mathcal{N}\left(\begin{bmatrix}\mu_\alpha \\ \mu_\beta\end{bmatrix}, \begin{bmatrix}\Sigma_{\alpha\alpha} & \Sigma_{\alpha\beta} \\ \Sigma_{\beta\alpha} & \Sigma_{\beta\beta}\end{bmatrix}\right) = \mathcal{N}^{-1}\left(\begin{bmatrix}\eta_\alpha \\ \eta_\beta\end{bmatrix}, \begin{bmatrix}\Lambda_{\alpha\alpha} & \Lambda_{\alpha\beta} \\ \Lambda_{\beta\alpha} & \Lambda_{\beta\beta}\end{bmatrix}\right)}$$

- Goal: compute marginalization or prediction calculation $\boldsymbol{p(\alpha) = \int p(\alpha, \beta) d\beta}$:

Covariance form: $\boldsymbol{\Sigma = \Sigma_{\alpha\alpha}}$.

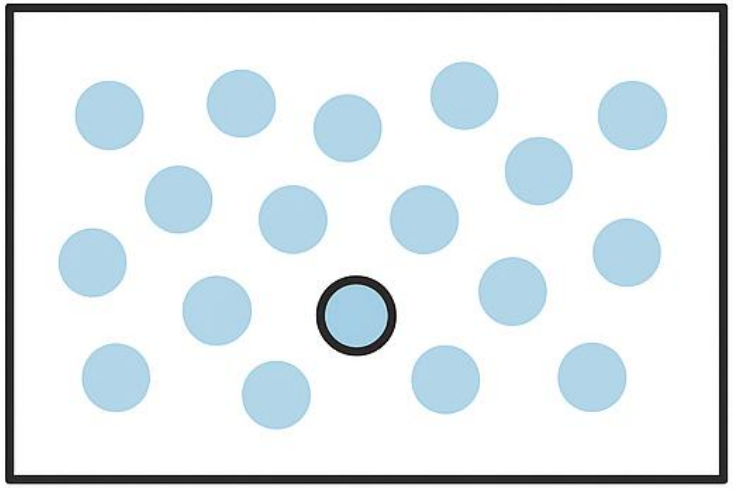
Information form: $\boldsymbol{\Lambda = \Lambda_{\alpha\alpha} - \Lambda_{\alpha\beta}\Lambda_{\beta\beta}^{-1}\Lambda_{\beta\alpha}}$.

Covariance matrix Information matrix

(Sagun et al. 2018)

Lee, J., Humt, M., Feng, J., & Triebel, R (2020). Estimating model uncertainty of neural networks in sparse information form. In *International Conference on Machine Learning* (pp. 5702-5713).

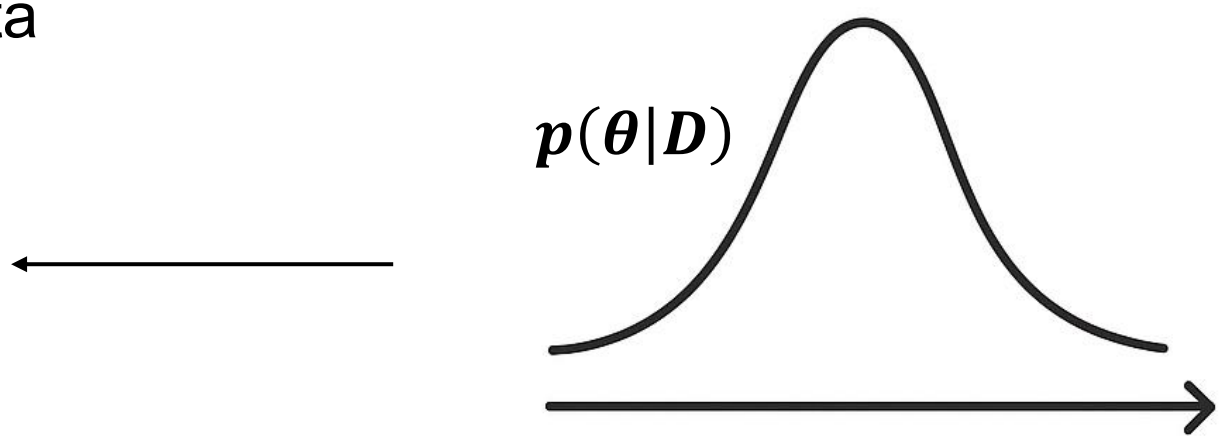
Evaluation: Pool-based Active Learning



Pool of unlabelled data

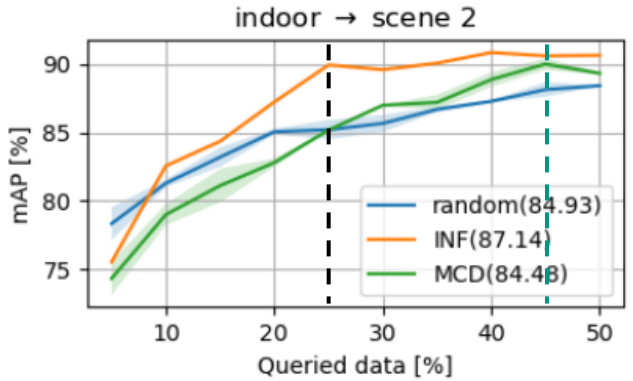
Goal: select the most informative data in order to reduce labelling costs.

Pick data that my model doesn‘t know!



Lee, J., Balachandran, R., Kondak, K., Coelho, A., De Stefano, M., Humt, M., ... & Triebel, R. (2024). Introspective Perception for Long-term Aerial Telemanipulation with Virtual Reality. *IEEE Transactions on Field Robotics*.

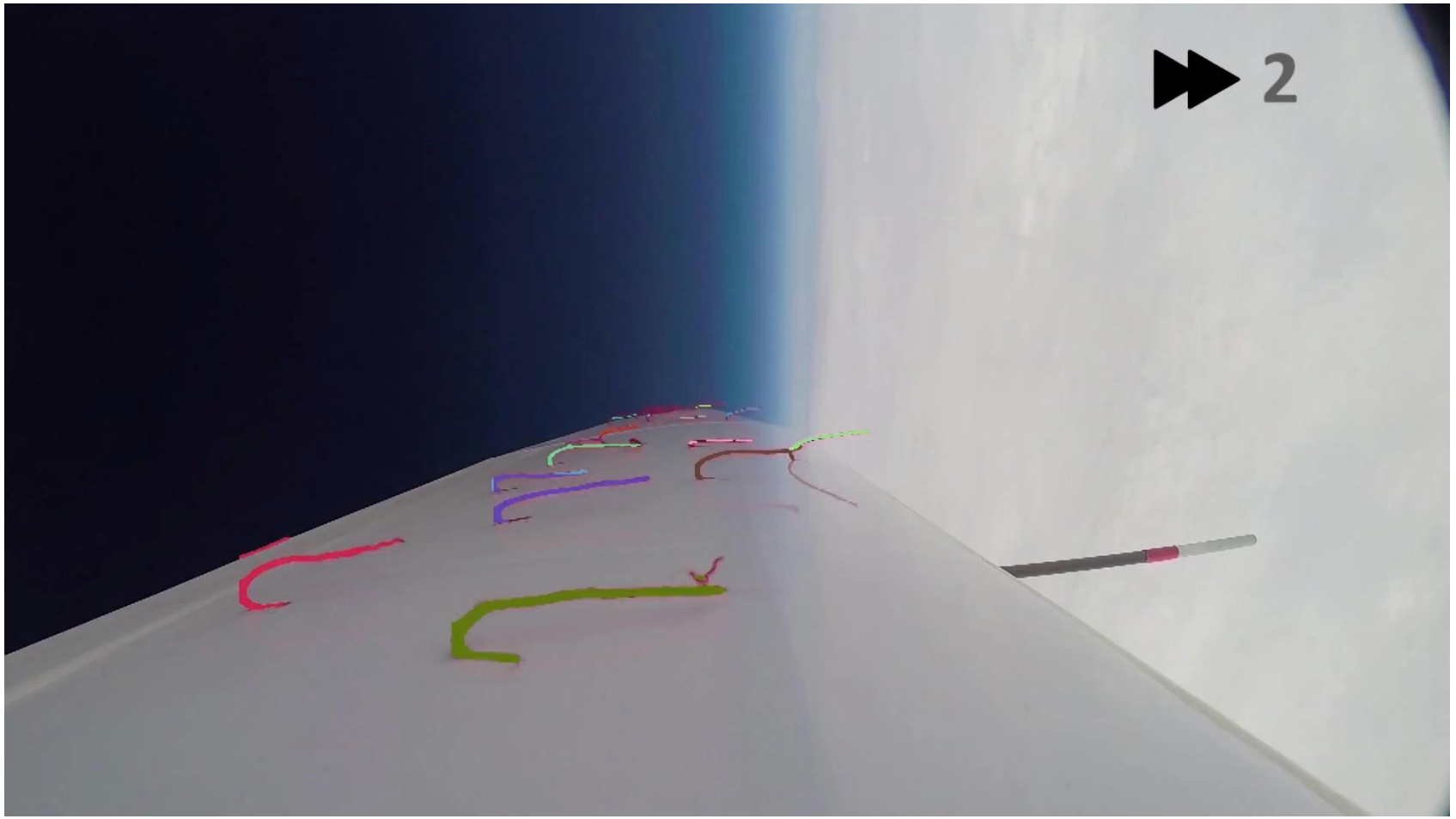
Outside lab: Reducing labelling costs



Expressive posterior → pool-based active learning performance.

Lee, J., Balachandran, R., Kondak, K., Coelho, A., De Stefano, M., Humt, M., ... & Triebel, R. (2024). Introspective Perception for Long-term Aerial Telemanipulation with Virtual Reality. *IEEE Transactions on Field Robotics*.

AI4Science: Reducing labelling costs

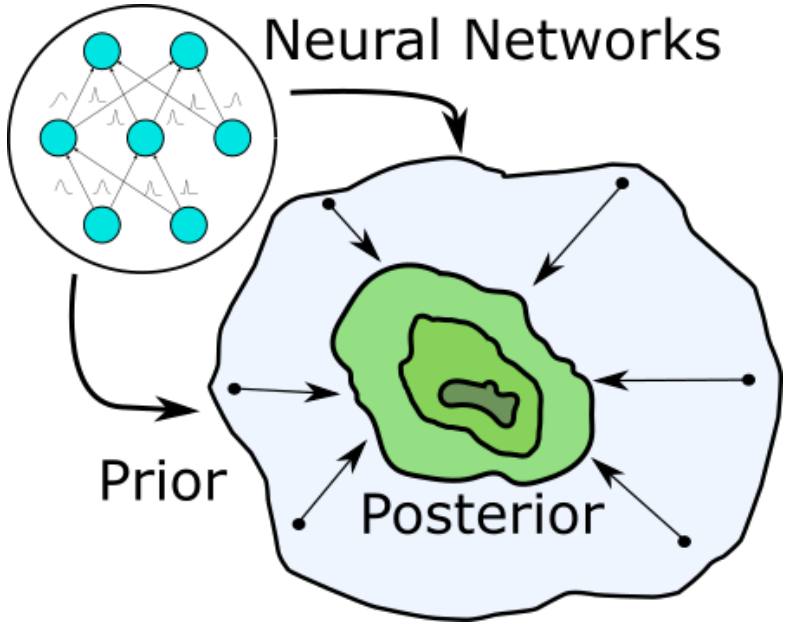


Lee, J*., Olsman, W. F. J.*, & Triebel, R. (2023). Learning fluid flow visualizations from in-flight images with tufts. *IEEE Robotics and Automation Letters*. *Co-first authors.

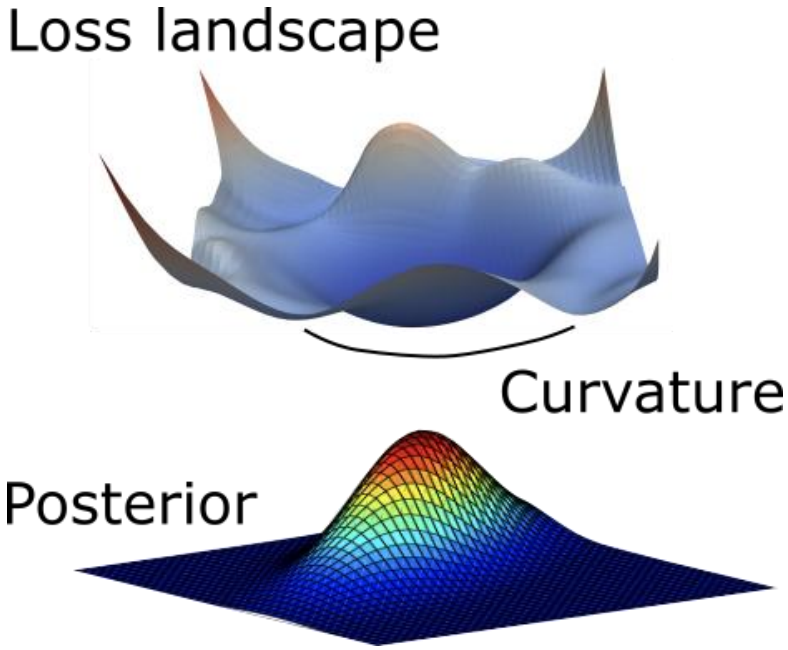

IBG
karmetal
MOTION
KUKA Innovation Award

Overview

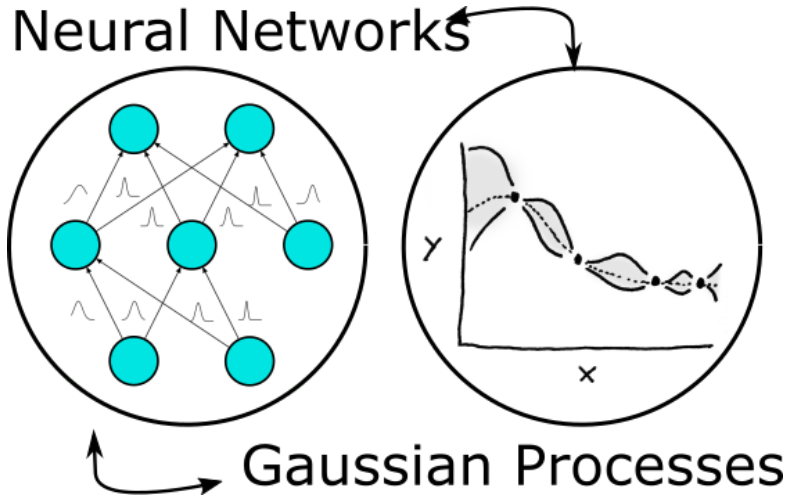
On Priors



On Posteriors



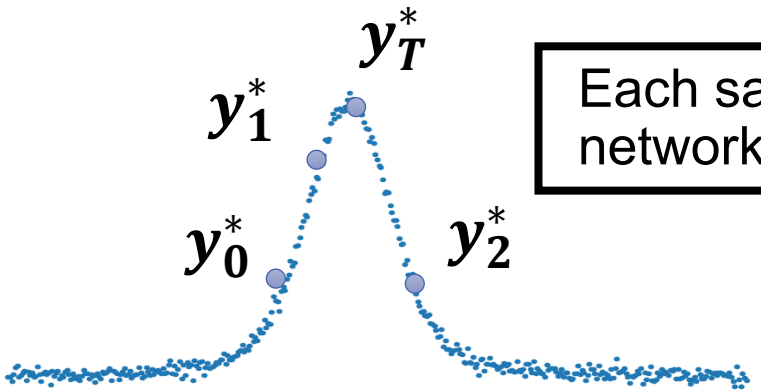
On Predictions



How to efficiently predict through marginalization by avoiding any sampling?

Challenge: Sampling is expensive

Sampling methods



Each sample = new network predictions!

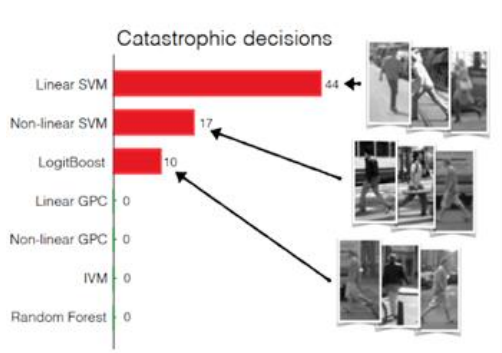
- Problem: Marginalization is intractable!

$$p(y^*|x^*, D) = \int p(y^*|\theta)p(\theta|D)d\theta$$

- Solution: Monte-carlo integration.

$$\approx \frac{1}{T}\sum_{t=1}^{T} f_{\theta=\theta_t}(x^*) \quad \text{for } \theta_t \sim p(w|D)$$

Sampling-free methods



(Grimmett et al, 2016 IJRR)

- Not every algorithm needs sampling.
- Gaussian Processes (GP) marginalized analytically and still golden standards.

Idea: Use GPs for Deep Learning?

Lee, J., Feng, J., Humt, M., Müller, M. G., & Triebel, R. (2022). Trust your robots! predictive uncertainty estimation of neural networks with sparse gaussian processes. In *Conference on Robot Learning* (pp. 1168-1179). PMLR.

Idea: Theory of Neural Tangent Kernel

Early works (1990s)

Radford Neal, "Priors for Infinite Networks", 1995.

- Pioneered the GP & neural network connection.
- Increasing width, single hidden layer and independent priors on neural network weights.
- Neural networks converge to GPs with Neural Tangent Kernel (NTK): $\mathbf{K}(\mathbf{x}, \mathbf{x}) = \mathbf{J}_f^{\mathrm{T}}(\boldsymbol{x})\mathbf{J}_f(\boldsymbol{x})$.

Recent breakthroughs

Modern extensions of Neal 1995.

- Multiple hidden layers.
  → J. Lee et al (2018).
  → Matthews et al (2018).
- Convolution layers.
  → Alonso et al (2019).
- Bayesian inference.
  → Khan et al (2019).
- Finite width.
  → Novak et al (2022).

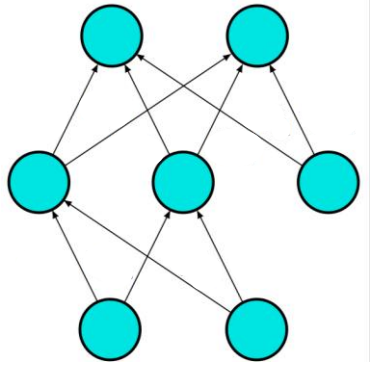
These works greatly advance the learning theory of deep learning!

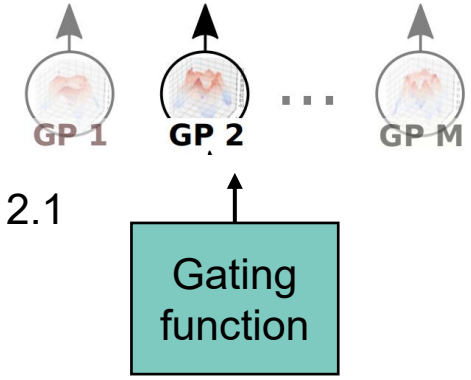
Lee, J., Feng, J., Humt, M., Müller, M. G., & Triebel, R. (2022). Trust your robots! predictive uncertainty estimation of neural networks with sparse gaussian processes. In *Conference on Robot Learning* (pp. 1168-1179). PMLR.

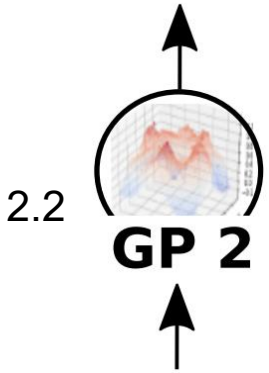
Idea: Theory of Neural Tangent Kernel

$$p(y^*|x^*, D) = \mathcal{N}\left(f_{\theta=\hat{\theta}}, \Sigma\right)$$

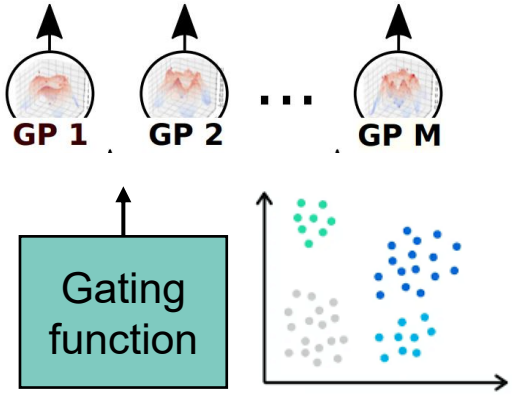
1. Compute neural network predictions: $y^* = f_{\theta=\hat{\theta}}(x^*)$.
2. Compute covariance Σ with mixtures of GP experts:
   2.1. Gating function assigns *m-th* GP expert.
   2.2. Using *m-th* GP expert: $\Sigma = K_{**} - K_*^T\left[K + \sigma_n^2 I\right]^{-1} K_*$.





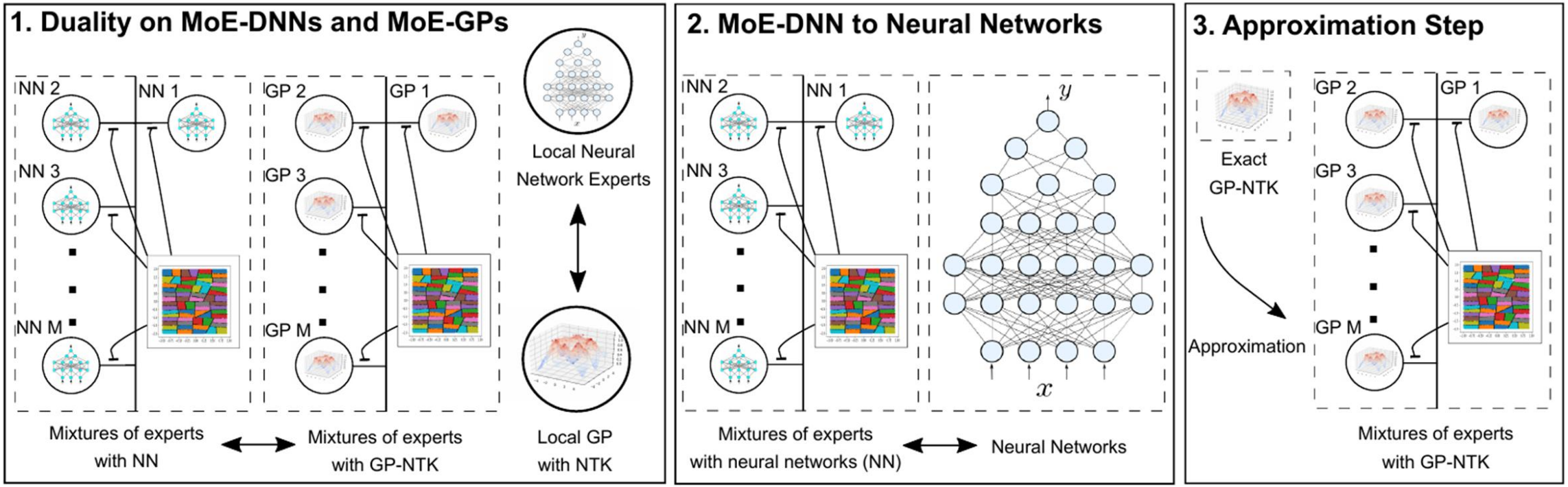


- \+ Sampling-free.
- \+ Winning combo.
- \+ Aids scaling.



Lee, J., Feng, J., Humt, M., Müller, M. G., & Triebel, R. (2022). Trust your robots! predictive uncertainty estimation of neural networks with sparse gaussian processes. In *Conference on Robot Learning* (pp. 1168-1179). PMLR.

Idea: Theory of Neural Tangent Kernel

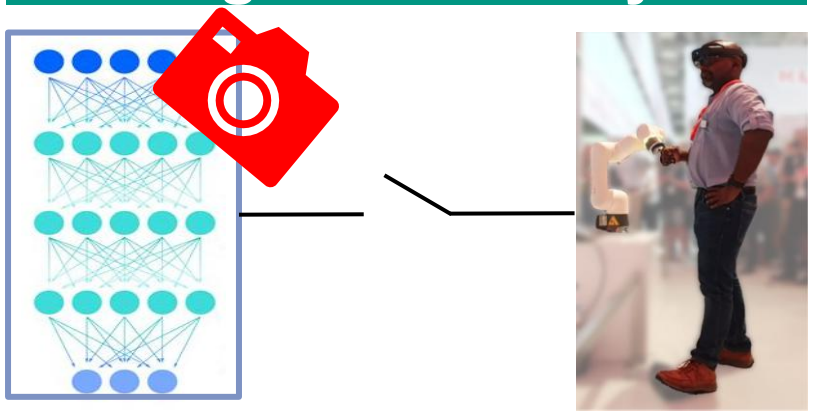


Contribution: theoretic foundation behind the proposed method with the NTK.

Lee, J., Feng, J., Humt, M., Müller, M. G., & Triebel, R. (2022). Trust your robots! predictive uncertainty estimation of neural networks with sparse gaussian processes. In *Conference on Robot Learning* (pp. 1168-1179). PMLR.

Evaluation: Perceptive Shared Autonomy

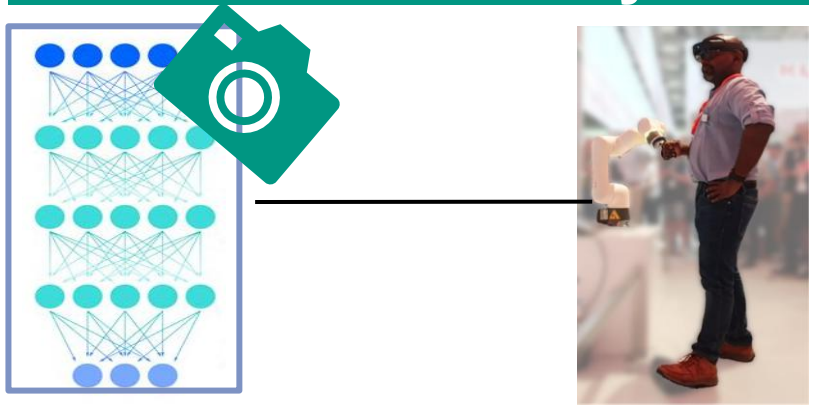
High uncertainty



High uncertainty

Level 0: teleoperation

Low uncertainty

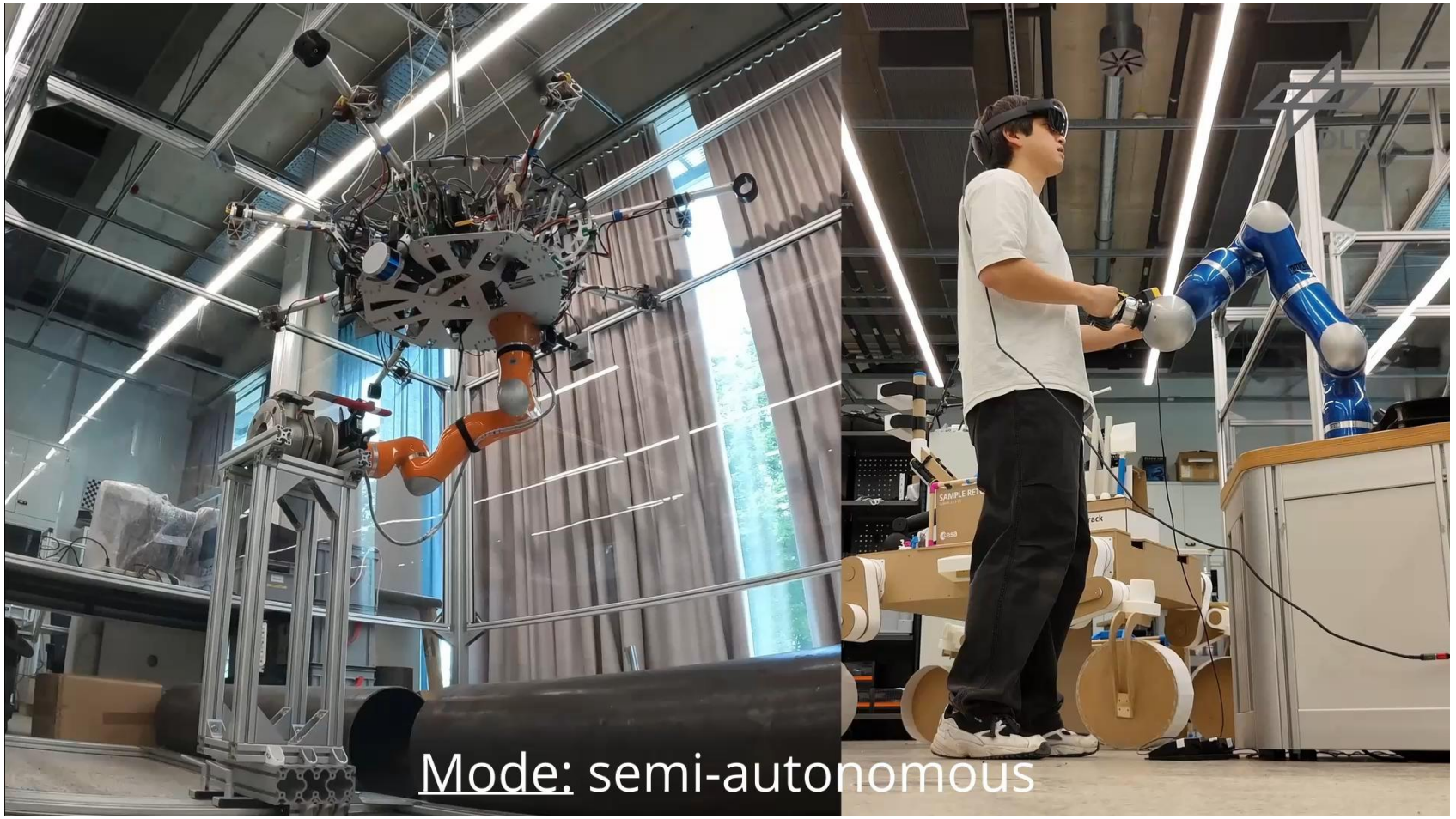


Low uncertainty

Level 3: assistance

SPIRIT: Perceptive Shared Autonomy for Aerial Manipulation under Uncertainty in Deep Learning. J. Lee et al. Under review. Finalist of KUKA Innovation Award 2023.

Robust Aerial Manipulation



Set 1	Vanila-teleop	SPIRITv2
Success rate [%] ↑	86.66±0.000	**100.0±0.000**
Time [s] ↓	160.46±144.1	**61.86±46.03**
Forces [N] ↓	11.66±1.325	11.72±1.414
Torques [Nm] ↓	6.674±0.748	7.442±0.810
NASA TLX [-] ↓	11.58±3.715	**7.422±2.613**
SUS score [-] ↑	51.00±25.73	**71.33±16.77**

Shared autonomy improves the performance.

Set 2	Vanila-VF	SPIRIT
Success rate [%] ↑	40.00±0.000	**100.0±0.000**
Time [s] ↓	335.0±202.2	**78.33±72.77**
Forces [N] ↓	13.70±1.018	**10.86±1.586**
Torques [Nm] ↓	7.202±0.449	**6.679±0.903**
NASA TLX [-] ↓	14.18±3.416	**8.288±4.137**
SUS score [-] ↑	35.83±20.92	**65.50±24.10**

Uncertainty-awareness improves system reliability.

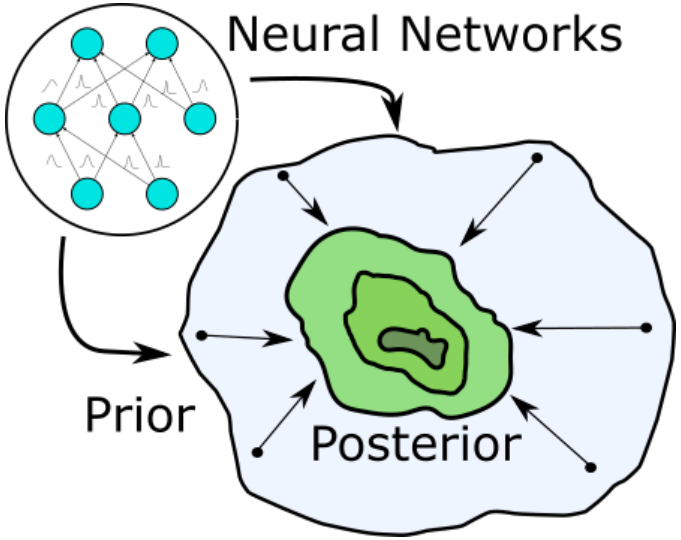
SPIRIT: Perceptive Shared Autonomy for Aerial Manipulation under Uncertainty in Deep Learning. J. Lee et al. Under review. Finalist of KUKA Innovation Award 2023.


iiQKA – Robots for the People
XITASO
Consulting. Development. Implementation.
Future-proofed production
XITASO
Unleash the potential of the Digital Twin
ROBOT MECHANICS
FOR EVERY CONTROLLER
KUKA

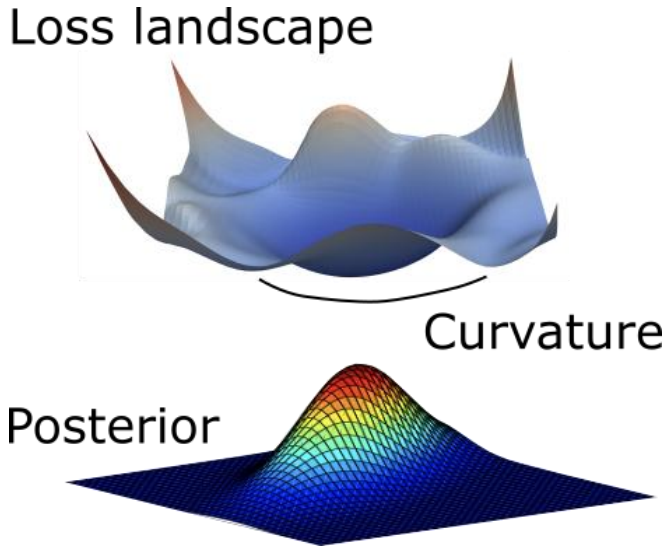
Conclusion

AI-based robotics that reason about uncertainty – no maximum likelihood.

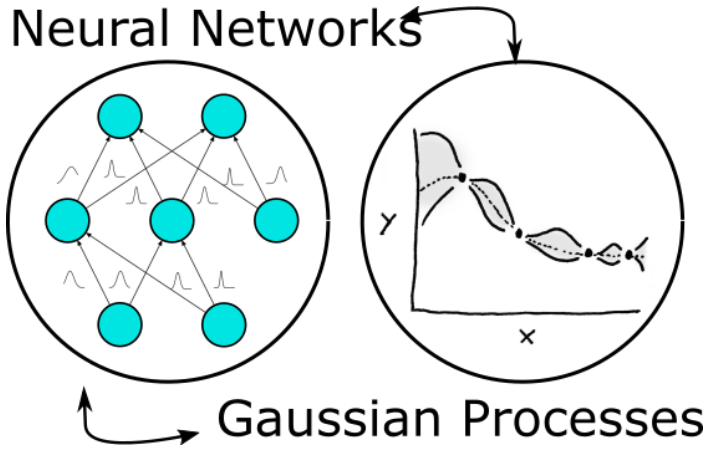
On Priors



On Posteriors



On Predictions



Novel probabilistic representations suitable for applications in robotics.

Achievements

- Tisby et al. 1989
- Mackay 1992
- Hinton and Van Camp 1993
- Neal 1995

AI winter

- Graves 2011
- Hernandez-Lobato & Adams 2015
- Gal and Ghahramani 2016
- Lakshminarayanan et al. 2017

When I started!

Promises:

- ⊕ Inclusion of domain knowledge via prior.
- ⊕ Quantifies uncertainty about the model.
- ⊕ Average different hypothesis about events.

Future work:

- **Prior:** Relevant data and task to learn the prior from.
  → Foundational priors that generalize more broadly.
- **Posterior:** Missing different modes of the posterior.
  → Combination with efficient variant of deep ensemble.
- **Predictions:** Limitations in scalability of GPs.
  → Hybrid methods for sparsification.
  → E.g., partitioning + inducing points + random features.

Dissemination

Research

10 > projects

ICRA 2020, **ICML 2020**, CoRL 2021, RA-L 2023**, **ICML 2023**, T-FR 2024,** RA-L2025a, RA-L2025b*

Thesis: 8 lead-author

9 lead author / 25 total *

Grants and Awards

Ministry of Trade, Industry and Energy

DLR

HELMHOLTZ GEMEINSCHAFT

Major industrial award*

Media Coverage

PRO ROBOTS

ProRobots

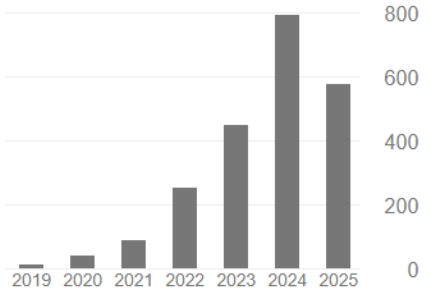
IEEE SPECTRUM

IEEE Video Friday

MIT Technology Review

MIT review

Academic Recognition



2400 > citations

Invited dissemination

*Out of those 1 lead / 4 co-author are under review / 2 of 9 co-lead.

*Finalist of KUKA Innovation Award 2023, 2024.

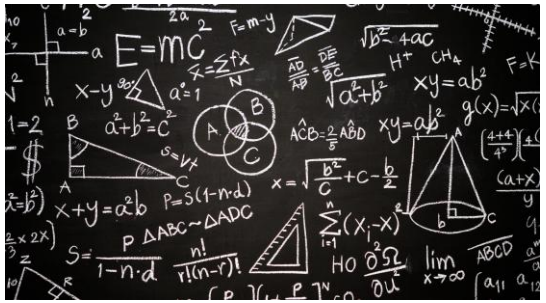
Dissemination

Theoretical contributions

ICRA2020, **ICML2020**, **CoRL2021**, RA-L2023**, **ICML2023****, T-FR2024, RA-L2025, RSS2026*

Thesis: 8 lead-author

9 lead author / 25 total *



10 key lemmas

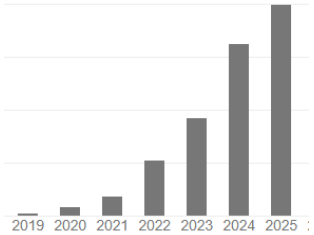
Technology transfer potential

Major industrial awards***

SAMSUNG, HD HYUNDAI, Chevron, slb, Vestas, SBM OFFSHORE, Deutsche Windtechnik

PI of market research project

Community Recognition



2700>citations

International dissemination

PRO ROBOTS, MIT Technology Review, IEEE SPECTRUM

Media coverages

Multidisciplinarity

5 perception tasks

7 different robots

6 outside-lab settings

*Out of those 1 lead / 2 co-author are under review.

**co-lead.

***Finalist of KUKA Innovation Award 2023, 2024.

Mentors and collaborators











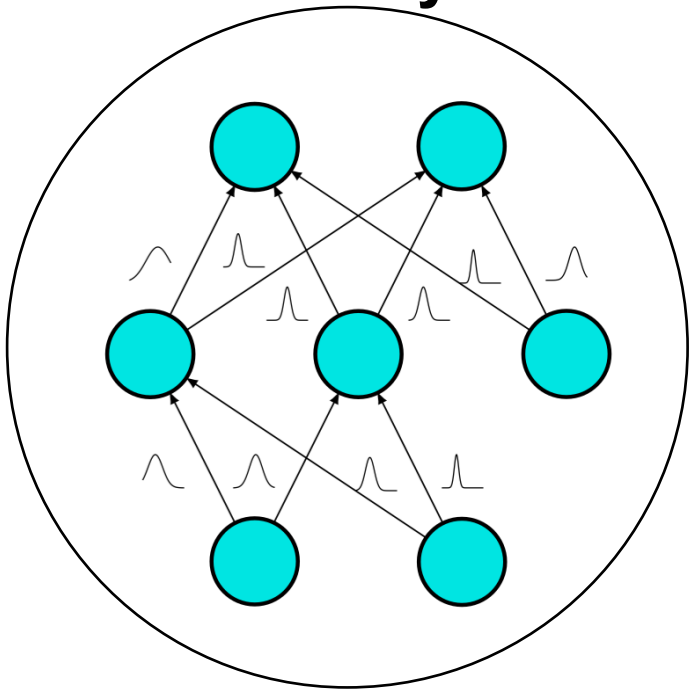

No Sisyphus.
No maximum likelihood!
Thank you for listening!

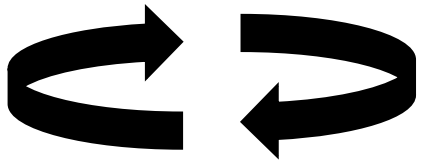
A: Overview

Major novelty

Uncertainty in Deep Learning: A Probabilistic Robotics Perspective

Theory





Practice

Priors, Posteriors and Predictions

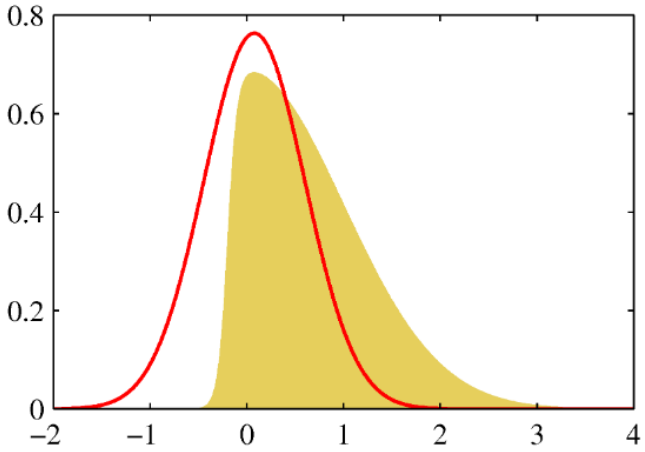
	Priors	Posteriors	Predictions
Research gap	Existing works choose uninformative prior despite the signs of prior misspecification.	Existing works simplifies the covariance matrix for scalability in lieu of expressivity.	Existing works require combining multiple predictions from the model's distribution.
Contributions	A novel method for learning scalable and structured posteriors of neural networks as information priors with generalization theory.	A novel information theoretic formulation that exploits the sparsity of inverse space of posteriors and its inference procedures.	A novel neural tangent kernel theory for sampling-free uncertainty estimation with mixtures of Gaussian Process experts.
Ramification	Domain knowledge can be incorporated for generalization and uncertainty estimation	More expressive but memory efficient approximation of posterior can be obtained	Real-time uncertainty estimation becomes possible with Gaussian Process formulation
Use cases	Excels in small data regime where the prior term dominates over the likelihood.	Excels in large data regime, large architectures but requires small memory consumption.	Excels in data regime upto 2 million. Permits large input but smaller output space.

Priors, Posteriors and Predictions

	Priors	Posteriors	Predictions
Accuracy	No design guarantees on preservation of accuracy (a-priori method).	Preserves accuracy of deep learning by design (post-hoc method).	Preserves accuracy of deep learning by design (post-hoc method).
Memory efficiency	Deployable on embedded GPUs. Memory grows with tasks	Deployable on embedded GPUs. Most memory efficient.	Deployable on embedded GPUs. Memory grows with data.
Run-time efficiency	Requires sampling for predictions and efficient only when parallelized.	Requires sampling for predictions and efficient only when parallelized.	Does not require sampling and thus efficient without parallelization.
Scalability	ImageNet-1K data-set within 24 hours on NVIDIA 1080 GPU.	ImageNet-1K data-set within 24 hours on NVIDIA 1080 GPU.	Two million data points within 15 hours on NVIDIA 1080 GPU.
Inferece scheme	Informative prior Laplace Approximation Sampling	Uninformative prior Laplace Approximation Sampling	Uninformative prior Laplace Approximation Gaussian Processes

Bayesian Deep Learning

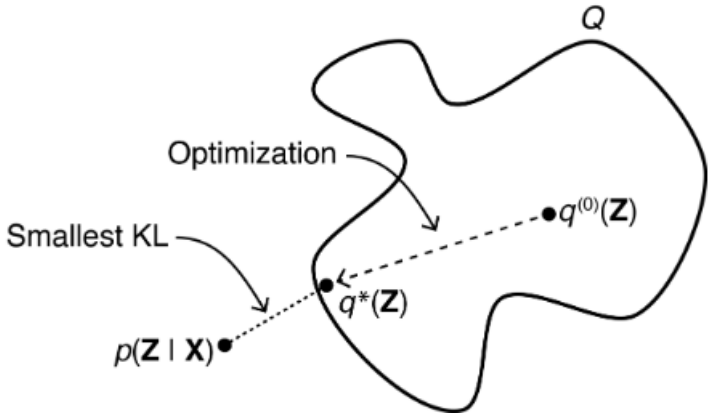
Laplace Approximation



Hessian approach

+ Computationally scalable.
+ Ease of implementation/training-free.
+ Competitive performance.
- Unimodal posterior.

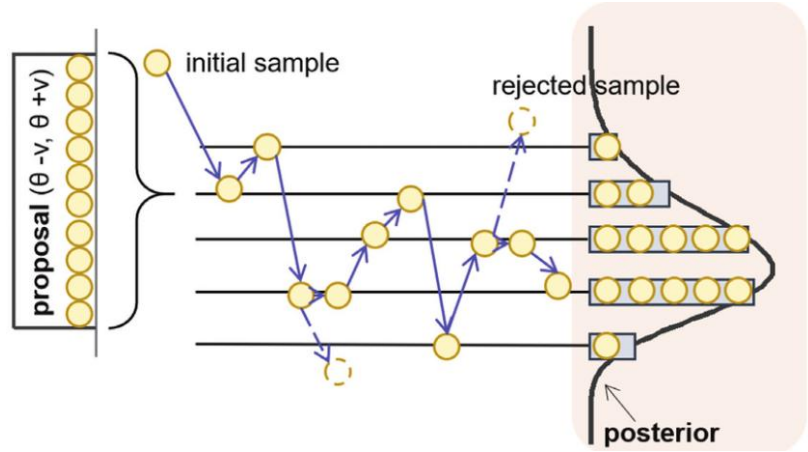
Variational Inference



Optimization approach

+ More scalable than sampling.
+ Theoretic gurarantees.
- Optimization challenges.

MCMC Sampling



Sampling approach

+ Best uncertainty estimates.
+ Theoretic gurarantees.
- Do not scale to large data and architectures.

★ More and more agreement that LA is practical alternative.
Post-hoc methods – rely on standard neural network training.

Why not Expectation Propagation?

Recap

Given posterior:

$$p(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{1}{p(\mathcal{D})} \prod_{i=1}^{M} f_i(\boldsymbol{\theta})$$

We want to obtain:

$$q(\boldsymbol{\theta}) = \frac{1}{Z} \prod_{i=1}^{M} \tilde{f}_i(\boldsymbol{\theta})$$

By minimizing:

$$\mathrm{KL}\left(\frac{1}{p(\mathcal{D})} \prod_{i=1}^{M} f_i(\boldsymbol{\theta}) \middle\| \frac{1}{Z} \prod_{i=1}^{M} \tilde{f}_i(\boldsymbol{\theta}) \right)$$

Application

We can formulate BNN posterior as:

$$p(W \mid D) \propto p(W) \prod_{n} p(y_n \mid x_n, W)$$

EP algorithm (roughly):

1. Initialize all factors
2. Pick a factor i to update
3. Compute tilted distribution
4. Approximate the factor by matching the moments with tilted distribution.
5. Get new q.
6. Repeat until convergence.

Why not?

1. Computing moment is not tractable.
2. Non-Gaussian likelihood of BNN with Gaussin likelihood.
3. Many iterations.
4. **Convergence issues.**

Assumed density filtering (PBP) is better but:

1. Accuracy is limited.
    1. Update once per factor.
    2. Mean-field approximation.
2. No factor revisiting – too fast shrinkage.
3. Ignores correlation between data points.

Easier to train neural networks with stochastic gradient descents.

Data and Model Uncertainty

Aleatoric Uncertainty	Epistemic Uncertainty
Uncertainty that comes from inherent noise in the data itself. **Irreducible (always there).**	Uncertainty that comes from lack of knowledge in the model. Reduced with more data.
Source: intrinsic randomness, ambiguity in labelling and label noises.	Source: limited or sparse training data, underrepresented regions in the input.

Model (epistemic) uncertainty is often more important than data uncertainty because it reveals when the model encounters **unknown or unexpected inputs**, allowing safer decisions and targeted learning.

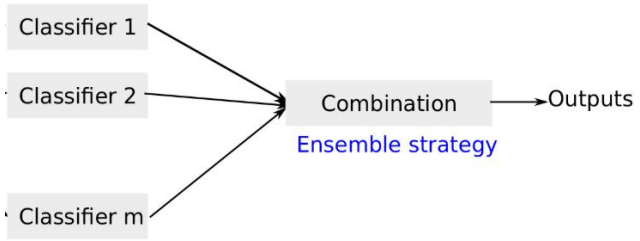
Uncertainty in Deep Learning

Conformal Predictions

Prediction sets guaranteed to contain the true outcome with a chosen probability. Conformal prediction instead produces a **set of plausible predictions** whose size reflects uncertainty. If the model is **uncertain**, the set is **larger** (e.g., {“cat”, “fox”, “dog”}).

+ Easy to use / model agnostic.
+ Guranteed coverage.
- Need for calibration data.
- Doesn‘t provide uncertainty.

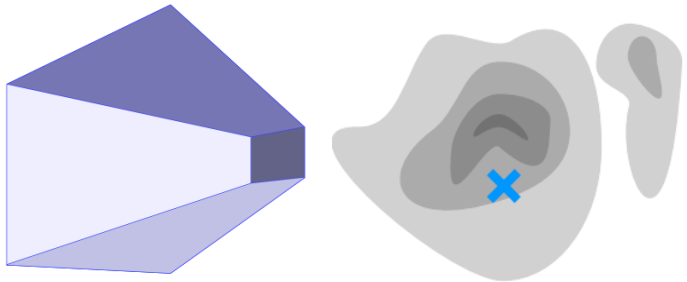
Deep Ensembles



+ Easy to use / model agnostic.
+ Superiority to MC-dropout.
- Computational costs.

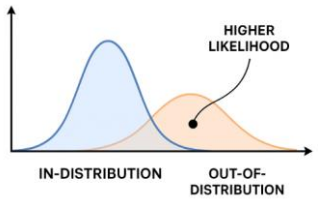
Note: deep ensemble can be combined with Bayesian models. * state-of-the-art.

OOD Detectors

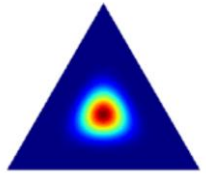


Latent space + generative model

+ Easy to use / model agnostic.
+ Fast at run-time.
- Doesn‘t provide uncertainty.
- Need good latent space.
- Likelihood paradox.



Evidential Learning



$\alpha = [10, 10, 10]$
$u = 0.1$

Network outputs parameters of dirichlet or norma-inverse-gamma distribution.

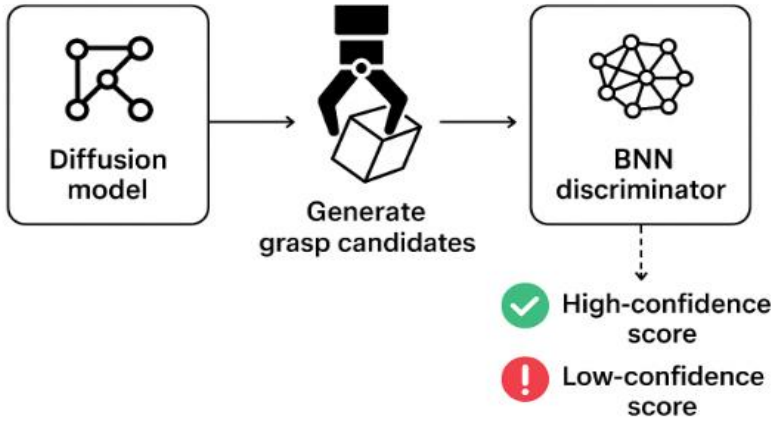
+ Fast at run-time.
+ Superiority to MC-dropout.
- Need to optimize / loss change.
- Need for out-of-distribution data.
- Indirect model uncertainty.

Use cases

LLM – CP.
YOLO – OOD detection.
Semantic segmentation – Evidential
Offline – DE + BNN

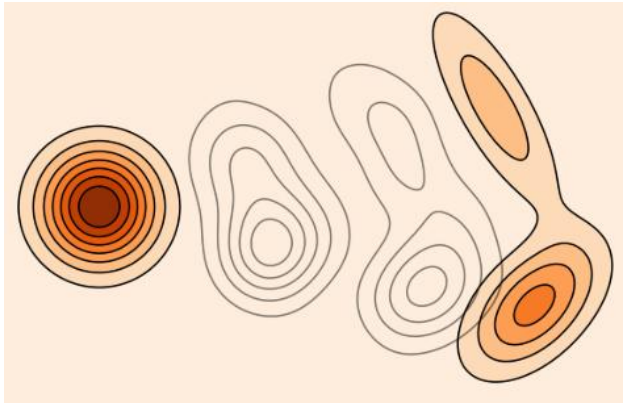
Uncertainty & GenAI

GenAI + BNN



- Diffusion model generate grasp candidates.
- A **discriminator** evaluates the quality of candidates.
- Likelihood != success probability (more to encode).

GenAI in Bayesian inference



- Diffusion model to generate samples from posterior.
- Likeihood-Free Bayesian inference to simulate data.
- **Flow models to capture multi-modal posterior.**

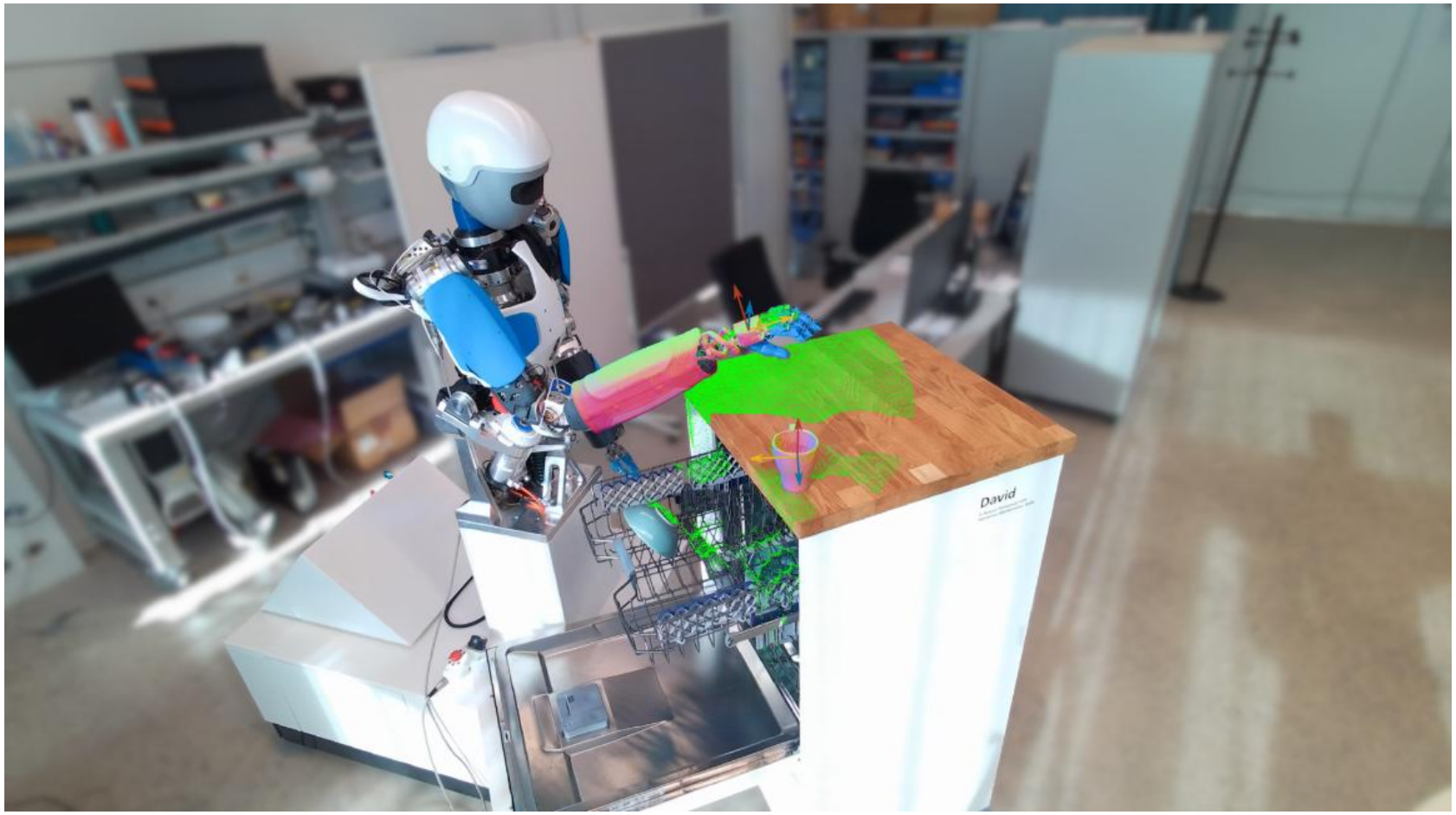
Visual foundational model

- Visual foundational models are discriminative models, trained like to generate.
- Masked Autoencoter (MAE) and Segment-anything is a discriminative model.

Rigorous Perception

Rigorous keywords:

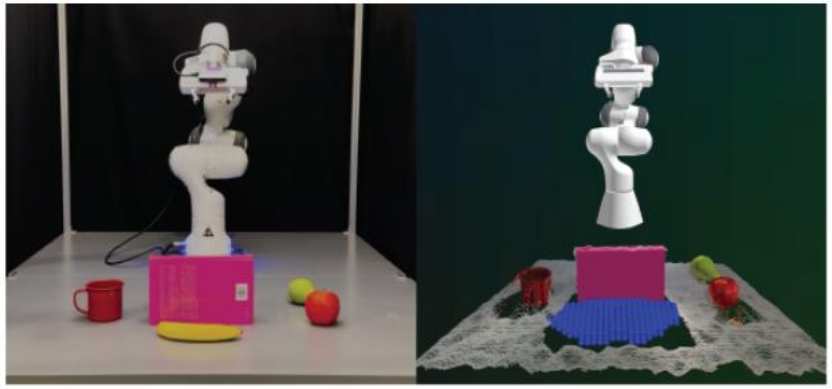
- Robust
- Interactive
- Interpretable
- Generalizable
- Omnimodal
- Resource-efficient
- Open
- Uncertainty-aware
- Semantic



B: Extensions

Task and Motion Planning 1

- Partially Observable Task and Motion Planning with Uncertainty and Risk Awareness (RSS 2024)



- If high uncertainty → gather information first.
- If low uncertainty → act directly.

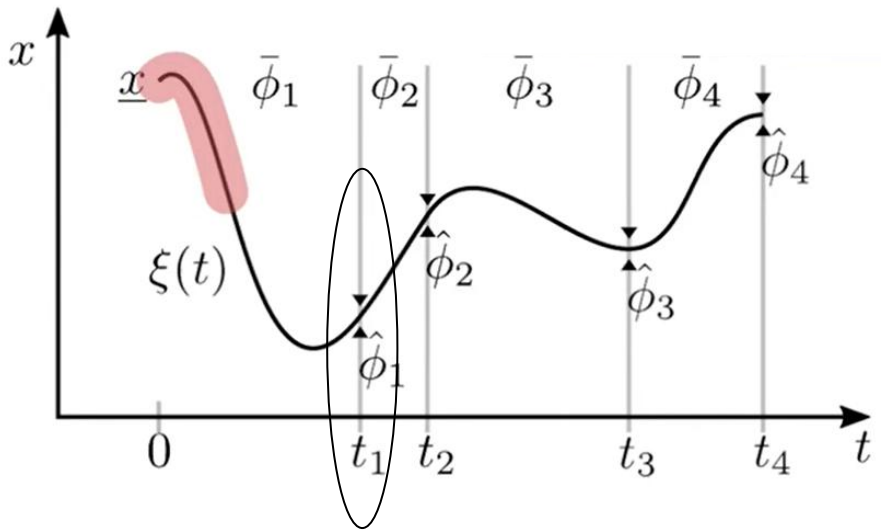
```
look_behind(box1):
    preconditions: can_see(box1)
    effects: reveal(object_location)
    uncertain_effects: {object_found, object_not_found}
```

Used Bayes3D for robot perception – 6D pose.

Authors: Aidan Curtis, George Matheos, Nishad Gothoskar, Vikash Mansinghka, Joshua Tenenbaum, Tomas Lozano-P ´ erez, Leslie Pack Kaelbling

Task and Motion Planning 2

- Deep Visual Constraints: Neural Implicit Models for Manipulation Planning from Visual Input (RAL2021)



- Task and motion planning with logic geometric programming.

$$\min_{\substack{x:[0,t_K]\to\mathfrak{X} \\ t_{1:K}}} \; t_K + \alpha \int_0^{t_K} c(\bar{x}(t))\, dt$$

$$\text{s.t. } \bar{x}(0) = \bar{x}_0,\ \dot{x}(t_K) = 0\,,$$
$$\forall_k : 0 < t_k < t_{k+1}\,,$$
$$\forall_k : \hat{\phi}_k(x(t_k)) \le 0,$$
$$\forall_{t\in[t_{k-1},t_k]} : \bar{\phi}_k(\bar{x}(t)) \le 0\,.$$

Optimal control with contraints.

- $h=\phi(q;V)$, where q is robot pose (or joint state) relative to object, and V the image views. Zero when feasible.
- This mapping is learned – Unet + MLP type architecture.

Rough idea: estimate uncertainty for contraints of neural networks.

- Chance contrained formulation: $\mu(x,u) + k_\alpha \sigma(x,u) \le 0$
- Probabilistic cost as soft contraints: $J = \sum_{t=0}^{H-1} \ell(x_t,u_t) + \lambda\left[\mu(x_t,u_t) + k_\alpha\sigma(x_t,u_t)\right]_+^2$

The probability of violating the contraint is bounded.

Optimization penalizes also the risky actions.

C: Important Details

Uncertainty Evaluation

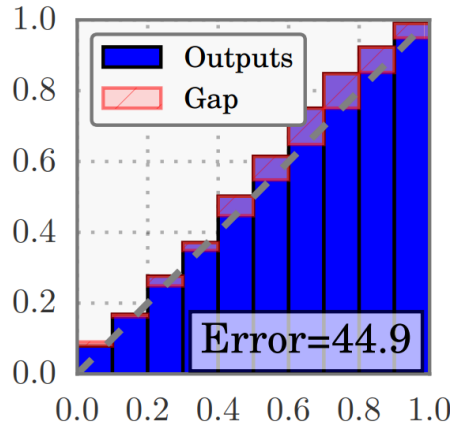
Tasks	Classification[1-3,5,7,10-13,16-19,21-25] & Regression[1-11,13,14,16,21,23] (incl. calibration, open set recognition, distribution shifts)
Downstream	Reinforcement learning[1,2,8,22], Active learning[4,7,13,18,22,23], Adversarial attacks[3,5,23], Bayesian optimization[8], Language modeling[11,17], Continual and transfer learning[12,22,24,25], Few-shot learning[18,25], Generative model[19], & Pruning[15]
Metrics	Test LL or NLL[1,4,6-13,16-18,20-26], ECE or Reliability diagram[7,11-13,17,18,20,22-26], Empirical CDF for entropy[3,5,7,12,18,21,23], AUROC[12,17-19,20,23-25], and Brier score [16,17,18,23] & Latency[20,21,23]
Dataset	Toy example[1-10,14,16,19,21], PLEX[18,25], CO2[1], Sarcos & KUKA[21], fashionMNIST[19,24,26], UCI[1,2,4,6-11,13,16,23,26], ImageNet[7,11,12,16-18,23], 20newgroup[17], Criteo Display Advertising Challenge[17], RETINA[18], CLINC OOS intent detection[20,23], CIFAR10[3,7,12,13,16-20,22-24], SHVN[7,16,19,20,24], CIFAR100[5,10,11,12,18,20,23,24,26], Wilds[22], Diabetic Retinoscopy Detection[23], Wikipedia Toxcity[23], MNIST[1-3,5,7,16,17,19,22,25,26], notMNIST[3,5,7,16], YCB[25] & GPT2[26], TinyStories[26]

Proper score rule

A scoring rule SSS is **proper** if the expected score is **maximized** when the predicted probabilities match the true probabilities.

$$\mathbb{E}_{Y\sim p}[S(p,Y)] \geq \mathbb{E}_{Y\sim p}[S(q,Y)] \quad \forall q$$

Calibration



Laplace Approximation

Neural Networks: $f_{\boldsymbol{\theta}} : \mathbb{R}^d \mapsto \mathbb{R}^k, \quad f_{\boldsymbol{\theta}}(\mathbf{x}) = \varphi(\boldsymbol{W}_L \varphi(\boldsymbol{W}_{L-1}(\cdots \varphi(\boldsymbol{W}_1 \mathbf{x}))))$ $\quad \boldsymbol{\theta} = [\boldsymbol{W}_1, ..., \boldsymbol{W}_L]$

Prior: $p(\boldsymbol{\theta}|\alpha) = \mathcal{N}(\boldsymbol{\theta}|\mathbf{0}, \alpha^{-1}\boldsymbol{I})$

Likelihood: $p(\mathcal{D}|\boldsymbol{\theta}, \beta) = \prod_{N}^{n=1} \mathcal{N}(\mathbf{y}_i|f_{\boldsymbol{\theta}}(\mathbf{x}_i)), \beta^{-1})$

Posterior: $p(\boldsymbol{\theta}|\mathcal{D}, \alpha, \beta) \propto p(\boldsymbol{\theta}|\alpha)p(\mathcal{D}|\boldsymbol{\theta}, \beta)$

Laplace Approximation – Posterior:

$$\ln p(\boldsymbol{\theta}|\mathcal{D}) \approx \ln p(\hat{\boldsymbol{\theta}}|\mathcal{D}) - \frac{1}{2}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T \boldsymbol{H} (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})$$

Taylor series around the mode

$$\frac{\partial^2}{\partial \boldsymbol{\theta}^2} \ln p(\boldsymbol{\theta}|\mathcal{D})\big|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}}$$
$$= \frac{\partial^2}{\partial \boldsymbol{\theta}^2} \ln p(\mathcal{D}|f_{\boldsymbol{\theta}})\big|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}} + \frac{\partial^2}{\partial \boldsymbol{\theta}^2} \ln p(\boldsymbol{\theta})\big|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}}$$
$$=: \boldsymbol{H}_{\text{likelihood}} + \boldsymbol{H}_{\text{prior}}.$$

Rewriting Hessian term

$$p(\boldsymbol{\theta}|\mathcal{D}) \approx \mathcal{N}(\boldsymbol{\theta}|\hat{\boldsymbol{\theta}}, \boldsymbol{H}^{-1})$$

Exponential on both sides and reverse engineering the density

Laplace Approximation

Same setting as before. Then, consider predictive distribution:

$$p(\mathbf{y}^*, |\mathbf{x}^*, \mathcal{D}) = \int p(\mathbf{y}^*|\mathbf{x}^*, \boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta}$$

$$\mathbb{E}_{\boldsymbol{\theta} \sim p(\boldsymbol{\theta}|\mathcal{D})}[p(\mathbf{y}^*|\mathbf{x}^*, \boldsymbol{\theta})] \approx \frac{1}{T} \sum_{t=1}^{T} p(\mathbf{y}^*|\mathbf{x}^*, \boldsymbol{\theta}^{(t)}) \quad \text{for} \quad \boldsymbol{\theta}^{(t)} \sim p(\boldsymbol{\theta}|\mathcal{D})$$

Taylor series approximation around the output.

$$f_{\boldsymbol{\theta}}(\mathbf{x}) \approx f_{\hat{\boldsymbol{\theta}}}(\mathbf{x}) + \boldsymbol{J}_f^T (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})$$

Rules of Gaussian – closure under marginalization.

$$p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}, \beta) \approx \mathcal{N}(\mathbf{y}|f_{\hat{\boldsymbol{\theta}}}(\mathbf{x}) + \boldsymbol{J}_f^T (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}), \beta^{-1})$$

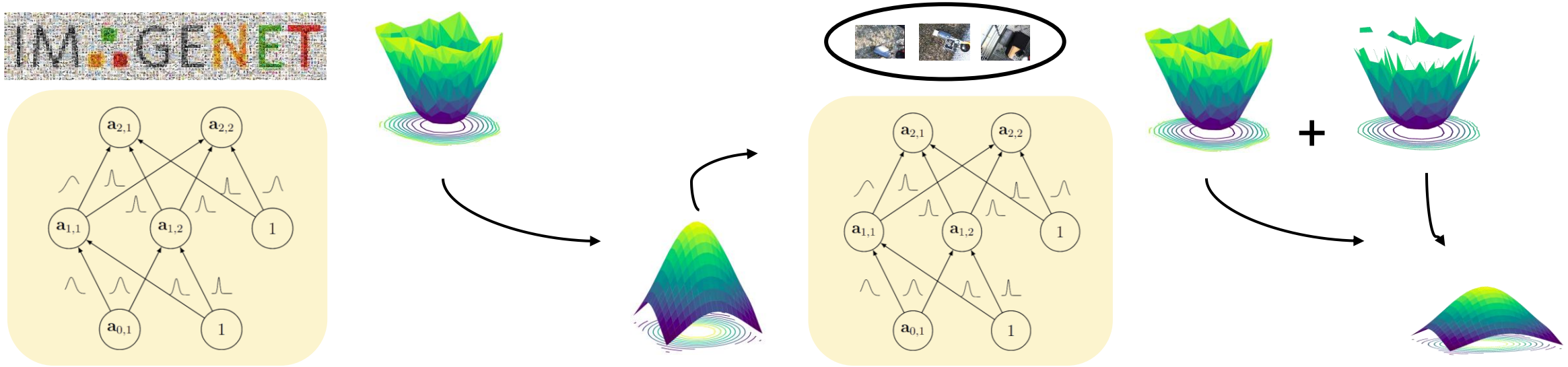
$$p(\mathbf{y}|\mathbf{x}, \mathcal{D}, \alpha, \beta) = \mathcal{N}(\mathbf{y}|f_{\hat{\boldsymbol{\theta}}}(\mathbf{x}), \beta^{-1} + \boldsymbol{J}_f^T \boldsymbol{H}^{-1} \boldsymbol{J}_f)$$

Assumptions

#	Assumption	Explanation
1	**Posterior is unimodal (one main peak)**	The approximation is centered at a single mode (the MAP). If there are multiple modes, the Gaussian can't capture the shape well.
2	**Log-posterior is approximately quadratic near the mode**	The method assumes the second-order Taylor expansion is accurate — i.e. higher-order terms are negligible.
3	**Posterior is well-peaked (not flat or heavy-tailed)**	The curvature (Hessian) around the mode should dominate; otherwise, the Gaussian spread misrepresents uncertainty.
4	**Negative Hessian of the log-posterior is positive definite**	Ensures the local curvature corresponds to a maximum (not a saddle point), and defines a valid covariance matrix.
5	**Sufficient sample size / asymptotic normality**	When data are large, the posterior tends to become Gaussian around the true parameter (Bernstein–von Mises theorem). Laplace works best in that regime.
6	**Continuity and differentiability**	The log posterior must be twice continuously differentiable with respect to parameters (needed for the Taylor expansion).

Prior

Posterior as prior



At task 0 (broad data-set and large architecture)

- Maximum a posteriori estimation
  $\widehat{\boldsymbol{\theta}}^{(0)} \in \operatorname{argmax} p\left(\boldsymbol{\theta}^{(0)} \middle| \boldsymbol{D}^{(0)}\right).$
- Laplace Approximation
  $p\left(\boldsymbol{\theta}^{(0)} \middle| \boldsymbol{D}^{(0)}\right) \approx \mathcal{N}\left(\boldsymbol{\theta}^{(0)} \middle| \widehat{\boldsymbol{\theta}}^{(0)}, (\boldsymbol{F}^{(0)})^{-1}\right).$
- Kronecker-factorized information matrix
  $\boldsymbol{F}^{(0)} = \boldsymbol{F}^{(0)}_{\text{likelihood}} + \boldsymbol{F}^{(0)}_{\text{prior}} = \boldsymbol{L}^{(0)} \otimes \boldsymbol{R}^{(0)} + \gamma \boldsymbol{I}.$

At task 1 (our robotic data of interest):

- Prior learned on task 0
  $\pi\left(\boldsymbol{\theta}^{(1)}\right) = \mathcal{N}\left(\widehat{\boldsymbol{\theta}}^{(0)}, \boldsymbol{F}^{(0)^{-1}}\right).$
- Posterior update on task 1
  $p\left(\boldsymbol{\theta}^{(1)} \middle| \boldsymbol{D}\right) = \mathcal{N}\left(\widehat{\boldsymbol{\theta}}^{(1)}, \left(\boldsymbol{F}^{(1)} + \boldsymbol{F}^{(0)}\right)^{-1}\right).$
- Kronecker-factorized information matrix
  $\boldsymbol{F}^{(1)} = \boldsymbol{L}^{(1)} \otimes \boldsymbol{R}^{(1)} + \boldsymbol{L}^{(0)} \otimes \boldsymbol{R}^{(0)} + \gamma \boldsymbol{I}.$

Sums-of-Kronecker product computations

Problem: sums-of-kronecker product as kronecker product

- Sums-of-kronecker product is not a kronecker product:

$$\boldsymbol{F} = \boldsymbol{L}^{(1)} \otimes \boldsymbol{R}^{(1)} + \boldsymbol{L}^{(0)} \otimes \boldsymbol{R}^{(0)} + \gamma \boldsymbol{I} \neq \hat{\boldsymbol{L}}^{(1)} \otimes \hat{\boldsymbol{R}}^{(1)}$$

$$\boldsymbol{F} = \boldsymbol{L} \otimes \boldsymbol{R} = \begin{pmatrix} l_{11}\boldsymbol{R} & \cdots & l_{1n}\boldsymbol{R} \\ \vdots & \ddots & \vdots \\ l_{m1}\boldsymbol{R} & \cdots & l_{mn}\boldsymbol{R} \end{pmatrix}$$

- But people still need Kronecker factorization for efficiency:

  Storage: $\boldsymbol{F} = \boldsymbol{L} \otimes \boldsymbol{R}, \quad \boldsymbol{F} \in \mathbb{R}^{pm \times qn}, \boldsymbol{L} \in \mathbb{R}^{m \times n}, \boldsymbol{R} \in \mathbb{R}^{p \times q}$

  Inversion: $(\boldsymbol{L} \otimes \boldsymbol{R})^{-1} = \boldsymbol{L}^{-1} \otimes \boldsymbol{R}^{-1}$

- Hence, prior works make assumptions that: $\boldsymbol{L} \otimes \boldsymbol{R} + \sqrt{\tau}\boldsymbol{I}_{m \times n} \otimes \sqrt{\tau}\boldsymbol{I}_{p \times q} = \left(\boldsymbol{L} + \sqrt{\tau}\boldsymbol{I}_{m \times n}\right) \otimes \left(\boldsymbol{R} + \sqrt{\tau}\boldsymbol{I}_{p \times q}\right) = \hat{\boldsymbol{L}} \otimes \hat{\boldsymbol{R}}$

Idea: power method converges to rank-1 optimal solution

- Sums-of-kronecker product as an optimization problem:

$$\hat{\boldsymbol{L}}, \hat{\boldsymbol{R}} \in \operatorname{argmin}_{L,R} \left| \sum_{k=1}^{K} \boldsymbol{L}^k \otimes \boldsymbol{R}^k - \boldsymbol{L} \otimes \boldsymbol{R} \right|_F$$

- Reorder the elements leads to rank 1 approximation problem:

$$\left| \sum_{k=1}^{K} \boldsymbol{L}^k \otimes \boldsymbol{R}^k - \boldsymbol{L} \otimes \boldsymbol{R} \right|_F = \left| \sum_{k=1}^{K} \operatorname{vec}\left(\boldsymbol{L}^k\right) \operatorname{vec}\left(\boldsymbol{R}^k\right)^T - \operatorname{vec}(\boldsymbol{L}) \operatorname{vec}(\boldsymbol{R})^T \right|_F$$

- Power method converges to rank-1 optimal solution.

$$\boldsymbol{R}^{(n)} \leftarrow \frac{\sum_{k=1}^{K} \left\langle \boldsymbol{L}^k, \boldsymbol{L}^{(n-1)} \right\rangle_F \boldsymbol{R}^k}{\left| \sum_{k=1}^{K} \left\langle \boldsymbol{L}^k, \boldsymbol{L}^{(n-1)} \right\rangle_F \boldsymbol{R}^k \right|_F}$$

↕ Proposed power iterations

$$\boldsymbol{L}^{(n)} \leftarrow \frac{\sum_{k=1}^{K} \left\langle \boldsymbol{R}^k, \boldsymbol{R}^{(n-1)} \right\rangle_F \boldsymbol{L}^k}{\left| \sum_{k=1}^{K} \left\langle \boldsymbol{R}^k, \boldsymbol{R}^{(n-1)} \right\rangle_F \boldsymbol{L}^k \right|_F}$$

Priors with generalization guarantees

PAC-Bayes Theorem:

$$P_{D\sim P^N}\left(\forall \rho \in \mathcal{M}, \rho \ll \pi : \mathbb{E}_{g\sim\rho}\left[L_P^l(g)\right] \leq \delta(\rho, \pi, D, \varepsilon)\right) \geq 1-\varepsilon$$

PAC posterior — PAC prior — True risk — Generalization bound

For example, McAllester bound states: $\delta(\rho, \pi, D, \varepsilon) = E_{g\sim\rho}\left[\hat{L}_D^l(g)\right] + \sqrt{\frac{\mathbb{KL}(\rho||\pi)+\ln\frac{2\sqrt{N}}{\varepsilon}}{2N}}$

Formulation (Humt et al., 2020)

$$p\left(\boldsymbol{\theta}^{(1)}\middle|\boldsymbol{D}^{(1)}\right)$$

$$= \mathcal{N}\left(\widehat{\boldsymbol{\theta}}^{(1)}, \left(\boldsymbol{F}^{(1)}+\boldsymbol{F}^{(0)}\right)^{-1}\right)$$

$$\rightarrow \mathcal{N}\left(\widehat{\boldsymbol{\theta}}^{(1)}, \boldsymbol{\tau}\left(\boldsymbol{\beta}\boldsymbol{F}^{(1)}+\boldsymbol{\alpha}\left(\boldsymbol{F}^{(0)}\right)\right)^{-1}\right)$$

Find $\boldsymbol{\tau}$, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ per layer?

Idea: PAC-Bayes for probabilistic inference (P. Germain et al, 2016)

$\pi \rightarrow \pi\left(\boldsymbol{\theta}^{(1)}\right)$ $\quad$ $\rho \rightarrow p\left(\boldsymbol{\theta}^{(1)}\middle|\boldsymbol{D}^{(1)}\right)$

Propose an approximate upper bound for empirical risk

$$\mathbb{E}_{g\sim\rho}\left[\hat{L}_D^l(g)\right] \leq \mathbb{E}_{\theta\sim\rho}\left[\frac{1}{N}\sum_{i=1}^{N} -\frac{\ln p(\boldsymbol{y}_i|\boldsymbol{x}_i, f_\theta)}{\ln 2}\right]$$

$$\approx \frac{-\ln p\left(\boldsymbol{D}\middle|f_{\hat{\theta}}\right) + \frac{1}{2}\sum_{l\in\{L\}}\tau\,\mathrm{tr}\left(\boldsymbol{F}^{(1)}+\left(\beta\boldsymbol{F}^{(1)}+\alpha\boldsymbol{F}^{(0)}\right)^{-1}\right)}{N\ln 2}$$

Algorithmic overview

Approximate Bayesian inference

1. Download your pre-trained posterior as prior.
2. Fine tune your network using that prior.
3. Apply Kronecker-factorized Laplace Approximation.
   3.1 Hyperparameter tuning with differentiable PAC-Bayes.
   3.2 Sums-of-Kronecker product computations.

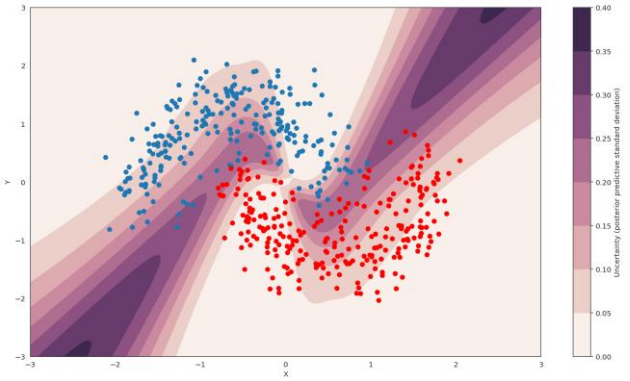
DLR-RM / BPNN Public

Code Issues Pull requests

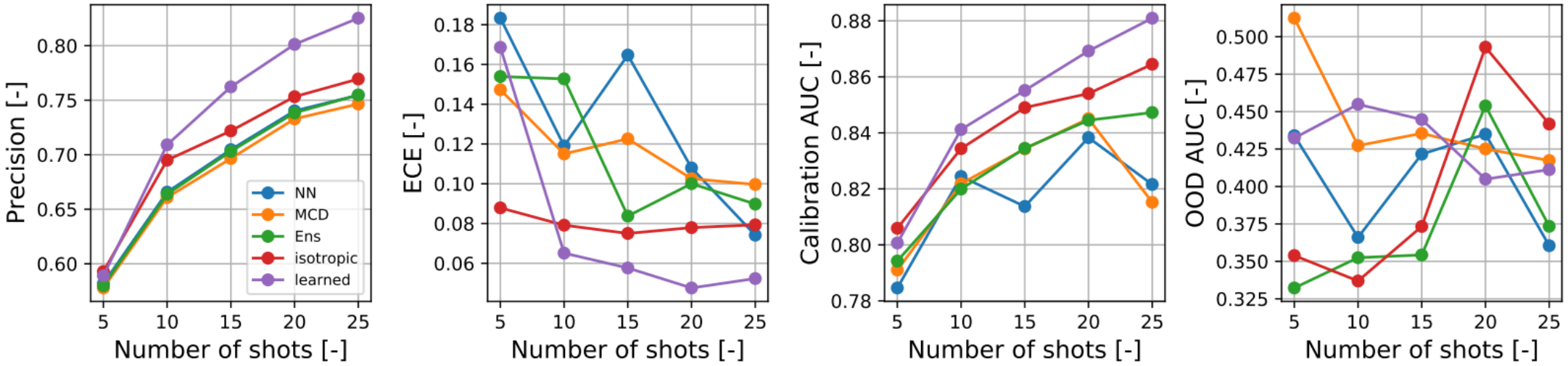
$$p\left(\boldsymbol{\theta}^{(1)}\middle|\boldsymbol{D}^{(1)}\right) = \mathcal{N}\left(\widehat{\boldsymbol{\theta}}^{(1)}, \left(\widehat{\boldsymbol{L}}^{(1)} \otimes \widehat{\boldsymbol{R}}^{(1)}\right)^{-1}\right).$$

Predictive distributions

$$p\left(\boldsymbol{y}^*\middle|\boldsymbol{D}^{(1)}, \boldsymbol{x}^*\right) = \int p\left(\boldsymbol{y}^*\middle|\boldsymbol{x}^*, \boldsymbol{\theta}^{(1)}\right) p\left(\boldsymbol{\theta}^{(1)}\middle|\boldsymbol{D}^{(1)}\right) d\boldsymbol{\theta}^{(1)}$$



Few-shot classification



Experiment setting

- Standardized benchmark from uncertainty baselines (Z. Nado et al, 2021).
- Average results over 8 data-sets.
- Few-shot image classification.

Key observations

- A working example learning prior from posteriors of Resnet-50 on ImageNet.
- Better generalization and uncertainty.
- Competitive on OOD detection.

Posterior

Inference in information form

- Approximate inference using Laplace Approximation:

$$\mathbf{p}(\boldsymbol{\theta}|\mathbf{D}) \sim \mathcal{N}\left(\boldsymbol{\theta}_{\mathbf{map}}, \mathbf{H}^{-1}\right) \quad \text{or} \quad \sim \mathcal{N}^{-1}\left(\boldsymbol{\eta}_{\mathbf{map}}, \mathbf{H}\right) *$$

- Employ EFB [George et al 2018] to estimate the Hessian H:

$$\mathbf{H} \approx \mathbf{I}_{\mathbf{efb}} = (\mathbf{U}_{\mathbf{A}} \otimes \mathbf{U}_{\mathbf{G}})\boldsymbol{\Lambda}(\mathbf{U}_{\mathbf{A}} \otimes \mathbf{U}_{\mathbf{G}})^{\mathbf{T}}$$ EFB Fisher Information matrix

$\mathbf{U}_{\mathbf{A}}$ is an eigenvector of $\mathbf{A} = \mathbb{E}[\mathbf{a}\mathbf{a}^{\mathbf{T}}]$ Forward pass

$\mathbf{U}_{\mathbf{G}}$ is an eigenvector of $\boldsymbol{G} = \mathbb{E}[\mathbf{g}\mathbf{g}^{\mathbf{T}}]$ Backward pass

$$\boldsymbol{\Lambda}_{\mathbf{ii}} = \mathbb{E}[\left((\mathbf{U}_{\mathbf{A}} \otimes \mathbf{U}_{\mathbf{G}})^{\mathbf{T}}\, \boldsymbol{\delta\theta}\right)_{\mathbf{i}}^{\mathbf{2}}]$$ Eigenvalue updates

***We omit the prior term for the simplicity of the presentation**

Inference in information form

- Diagonal elements of true Information matrix is known and easy to compute!

$\mathbf{I} = \mathbb{E}\left[\boldsymbol{\delta\theta}\boldsymbol{\delta\theta}^{\mathbf{T}}\right]$ by definition, and $\mathbf{I_{ii}} = \mathbb{E}\left[\boldsymbol{\delta\theta}_{\mathbf{i}}^{\mathbf{2}}\right]\ \forall \mathbf{i}$ Not true for the covariance matrix

- Resulting Kronecker-factored Eigen-decomposition plus diagonal structured information matrix:

$\mathbf{I_{inf}} = (\mathbf{U_A} \otimes \mathbf{U_G})\boldsymbol{\Lambda}(\mathbf{U_A} \otimes \mathbf{U_G})^{\mathbf{T}} + \mathbf{D^*}$ Exact on the diagonals

- This step brings a theoretical guarantee on improvements:

Lemma 1: theoretical guarantees regardless of the chosen data-set and architecture
Let I be the real information matrix, and let $\mathbf{I_{inf}}$ and $\mathbf{I_{efb}}$ be the INF and EFB estimates of it respectively.
Then, it is guaranteed to have $\|\mathbf{I} - \mathbf{I_{efb}}\|_{\mathbf{F}} \geq \|\mathbf{I} - \mathbf{I_{inf}}\|_{\mathbf{F}}$

Low rank sampling computations

- Computing the predictive uncertainty requires samples from the posterior

$$\mathbf{p}(\mathbf{y}^*|\mathbf{x}^*, \mathbf{D}) = \int \mathbf{p}(\mathbf{y}^*|\mathbf{x}^*, \boldsymbol{\theta})\mathbf{p}(\boldsymbol{\theta}|\mathbf{D})\mathbf{d}\boldsymbol{\theta}$$

- A naive approach is not sufficient if there are many parameters (e.g. millions)

$$\approx \frac{\mathbf{1}}{\mathbf{T}} \sum_{\mathbf{t=1}}^{\mathbf{T}} \mathbf{y}^*(\mathbf{x}^*, \boldsymbol{\theta}_\mathbf{t}^\mathbf{s}) \quad \text{for} \quad \boldsymbol{\theta}_t^s \sim \mathbf{p}(\boldsymbol{\theta}|\mathbf{D})$$

1. Evaluate the matrix:

$$\mathbf{I_{inf}} = (\mathbf{U_A} \otimes \mathbf{U_G})\boldsymbol{\Lambda}(\mathbf{U_A} \otimes \mathbf{U_G})^\mathbf{T} + \mathbf{D}$$

O(N2): infeasible

2. Perform Cholesky decomposition:

$$\mathbf{I_{inf}^{-1}} = \mathbf{F_c}\mathbf{F_c^T}$$

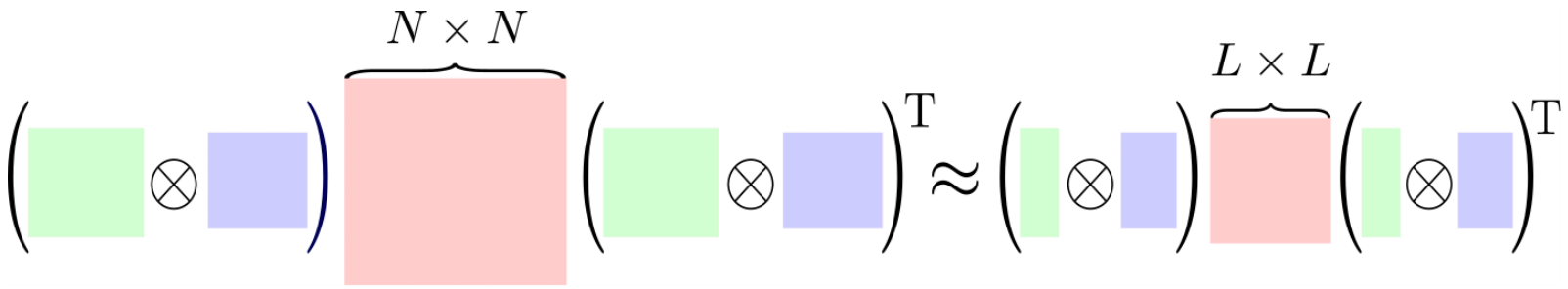
O(N3): infeasible

3. Draw samples from the distribution:

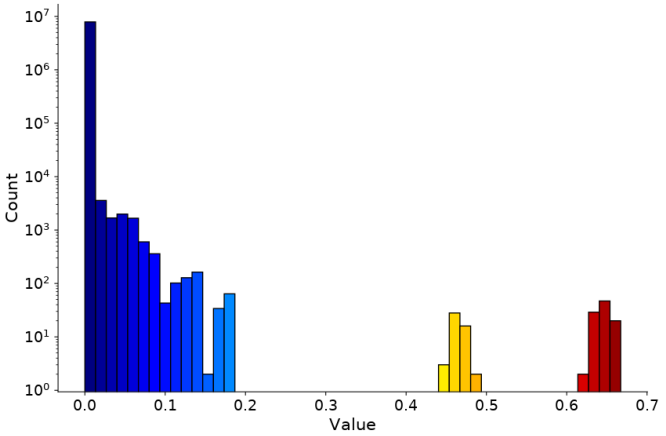
$$\boldsymbol{\theta}_\mathbf{t}^\mathbf{s} = \boldsymbol{\theta}_\mathbf{MAP} + \mathbf{F_c}\mathbf{X}^\mathbf{l}$$ with $\mathbf{X}^\mathbf{l}$ the samples of a standard Gaussian

Low rank sampling computations

- **Step 1:** low rank approximation while saving Kronecker products in eigenvectors:



$$(\mathbf{U_A} \otimes \mathbf{U_G})\boldsymbol{\Lambda}(\mathbf{U_A} \otimes \mathbf{U_G})^{\mathbf{T}} \approx (\mathbf{U_a} \otimes \mathbf{U_g})\boldsymbol{\Lambda}_{\mathbf{1:L}}(\mathbf{U_a} \otimes \mathbf{U_g})^{\mathbf{T}} *$$



- **Step 2:** samples with much lower cost and insignificant errors!

$$\boldsymbol{\theta}_{\mathbf{t}}^{\mathbf{s}} = \boldsymbol{\theta}_{\mathbf{MAP}} + \mathbf{F_c}\mathbf{X}^{\mathbf{l}}$$

$$\mathbf{F_c} = \mathbf{D}^{-\frac{\mathbf{1}}{\mathbf{2}}}\left(\mathbf{I_{nm}} - \mathbf{D}^{-\frac{\mathbf{1}}{\mathbf{2}}}(\mathbf{U_a} \otimes \mathbf{U_g})\boldsymbol{\Lambda}_{\mathbf{1:L}}^{\frac{\mathbf{1}}{\mathbf{2}}}\left(\mathbf{C}^{-\mathbf{1}} + \mathbf{V_s^T V_s}\right)^{-\mathbf{1}}\boldsymbol{\Lambda}_{\mathbf{1:L}}^{\frac{\mathbf{1}}{\mathbf{2}}}(\mathbf{U_a} \otimes \mathbf{U_g})^{\mathbf{T}}\mathbf{D}^{-\frac{\mathbf{1}}{\mathbf{2}}}\right)$$

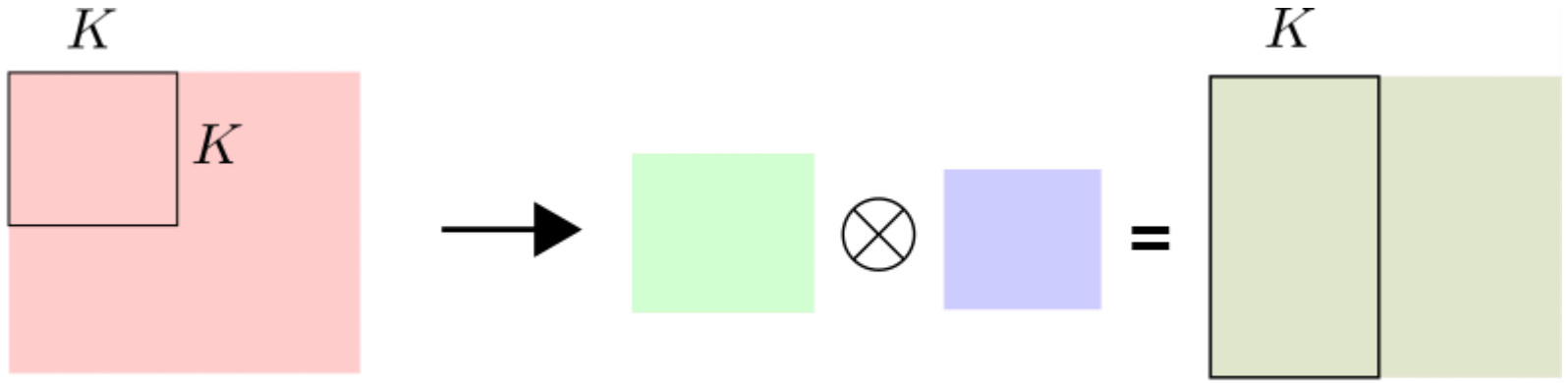
*Differs from $(\mathbf{U_A} \otimes \mathbf{U_G})_{\mathbf{1:L}}\boldsymbol{\Lambda}_{\mathbf{1:1}}(\mathbf{U_A} \otimes \mathbf{U_G})^{\mathbf{T}}$

Sparsification algorithm

- How to perform low rank approximation on the Kronecker-factored eigendecomposition?

$$(\mathbf{U_A} \otimes \mathbf{U_G})\mathbf{\Lambda}(\mathbf{U_A} \otimes \mathbf{U_G})^{\mathbf{T}} \approx (\mathbf{U_a} \otimes \mathbf{U_g})\mathbf{\Lambda_{1:L}}(\mathbf{U_a} \otimes \mathbf{U_g})^{\mathbf{T}}$$

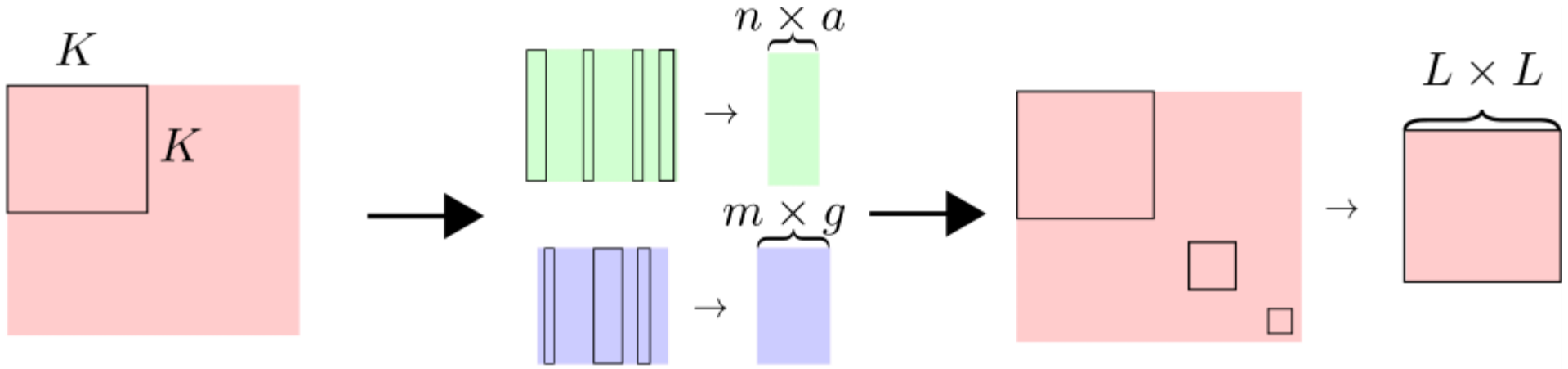
- Conventional low rank approximation such as singular value decomposition:



1. Find K top eigenvalues. 2. Select corresponding eigenvectors.

1. Select the top L eigenvalues and then: $\Lambda \approx \Lambda_{1:L}$
2. Using the indices of L eigenvalues, $V = (U_A \otimes U_G)$ and $V \approx V_{1:L}$

Sparsification algorithm



1. Find K top eigenvalues. 2. Select corresponding eigenvectors. 3. Select remaining eigenvalues.

Results: ImageNet

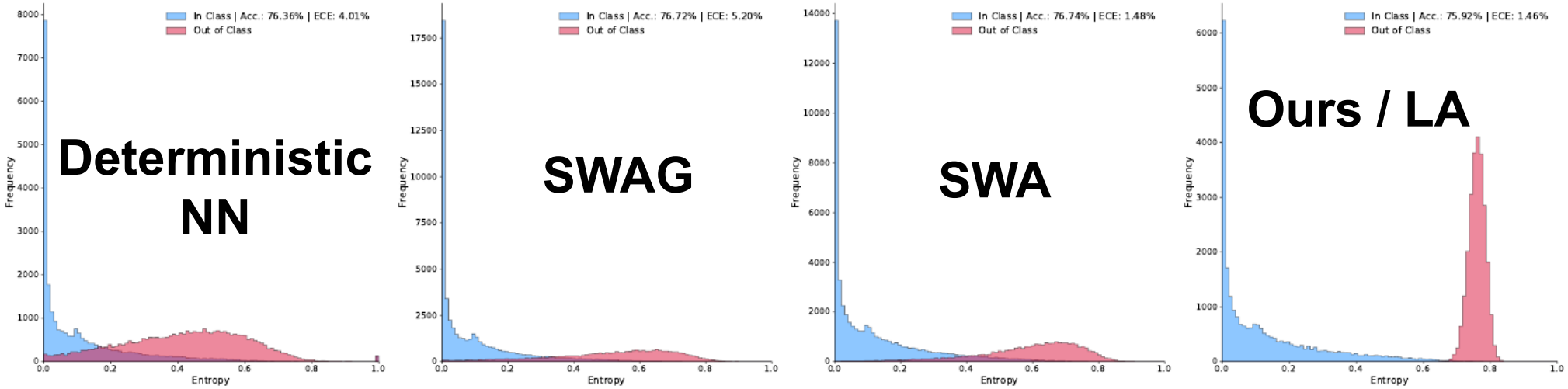


Table 3. **Network Space Complexity Comparison:** The total number of information matrix parameters and its size in MB are reported for ResNet and DenseNet variants. Lower the better. Here, we also check if the methods take into account the weight correlations (corr).

	Diag			KFAC			EFB			INF		
Model	#Parameters	Size	Corr	#Parameters	Size	Corr	#Parameters	Size	Corr	#Parameters	Size	Corr
ResNet18	11,679,912	*44.6*	X	95,013,546	362.4	✓	106,693,458	407.0	✓	12,317,373	**47.0**	✓
ResNet50	25,503,912	*97.3*	X	153,851,562	586.9	✓	179,355,474	684.2	✓	27,614,896	**105.3**	✓
ResNet152	60,041,384	*229.0*	X	389,519,018	1485.9	✓	449,560,402	1714.9	✓	65,558,402	**250.1**	✓
DenseNet121	7,895,208	*30.1*	X	103,094,954	393.3	✓	110,990,162	423.4	✓	9,711,081	**37.0**	✓
DenseNet161	28,461,064	*108.6*	X	379,105,514	1446.2	✓	407,566,578	1554.7	✓	32,329,191	**123.3**	✓

Predictions

Idea: Theory of Neural Tangent Kernel

Preliminaries

- Mixtures of experts (MoE):
  - M learners: $f_m(x)$
  - A gating function: $g_m(x)$
- A strict division of data: $y = \sum_{m=1}^{M} g_m(x) f_m(x)$

Either 0 or 1

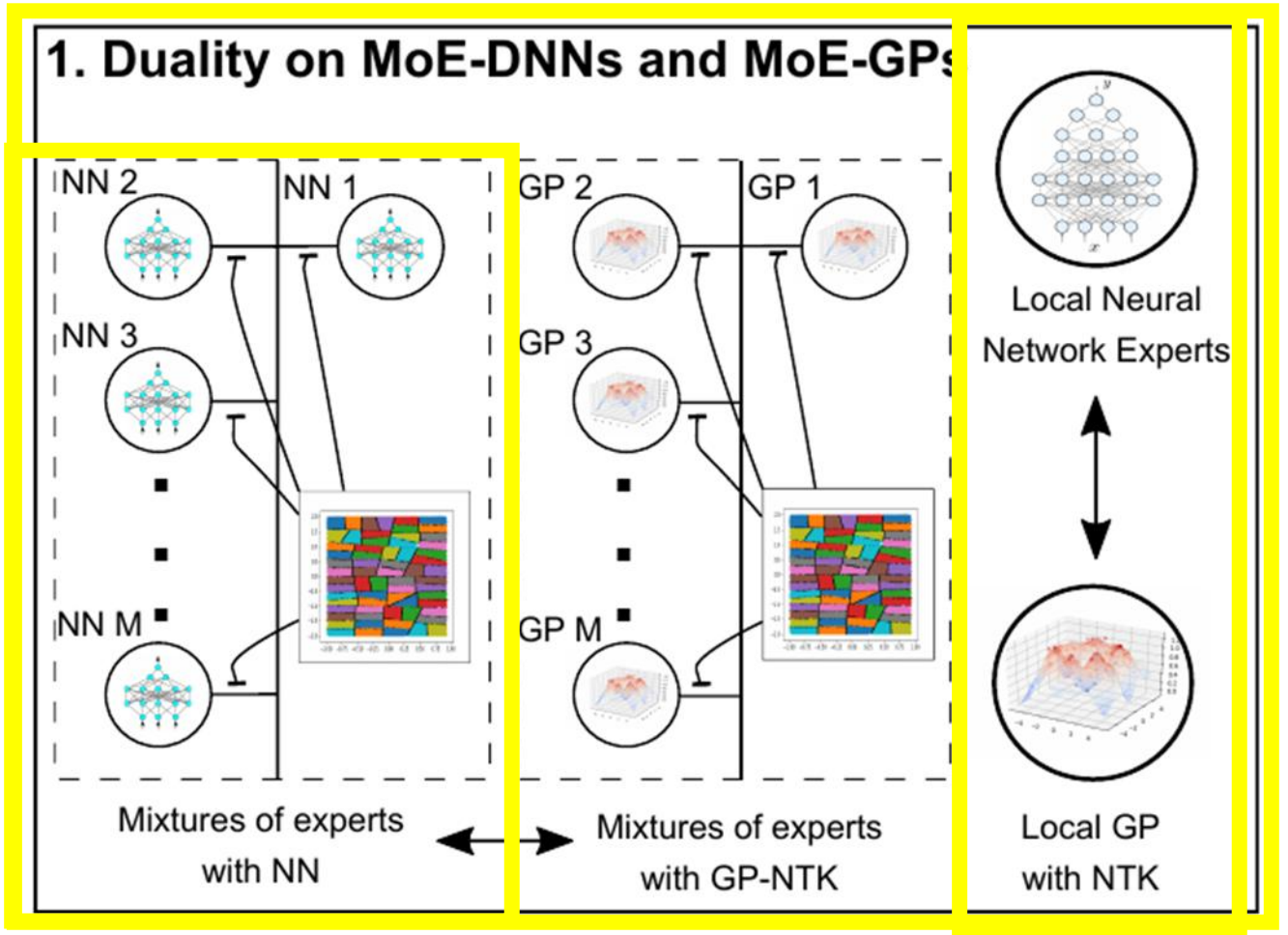
Bayesian duality

- Local NN ↔ Local GPs with NTK:

  $p(\theta_m; D_m) \leftrightarrow p(f_m; \widetilde{D_m})$

- Direct application of Khan et al (2019).
- MoE NN ↔ MoE GP with NTK:

  $\prod p(\theta_m; D_m) \leftrightarrow \prod p(f_m; \widetilde{D_m})$



Note: NN is any neural networks with valid Jacobians!

Idea: Theory of Neural Tangent Kernel

Problem

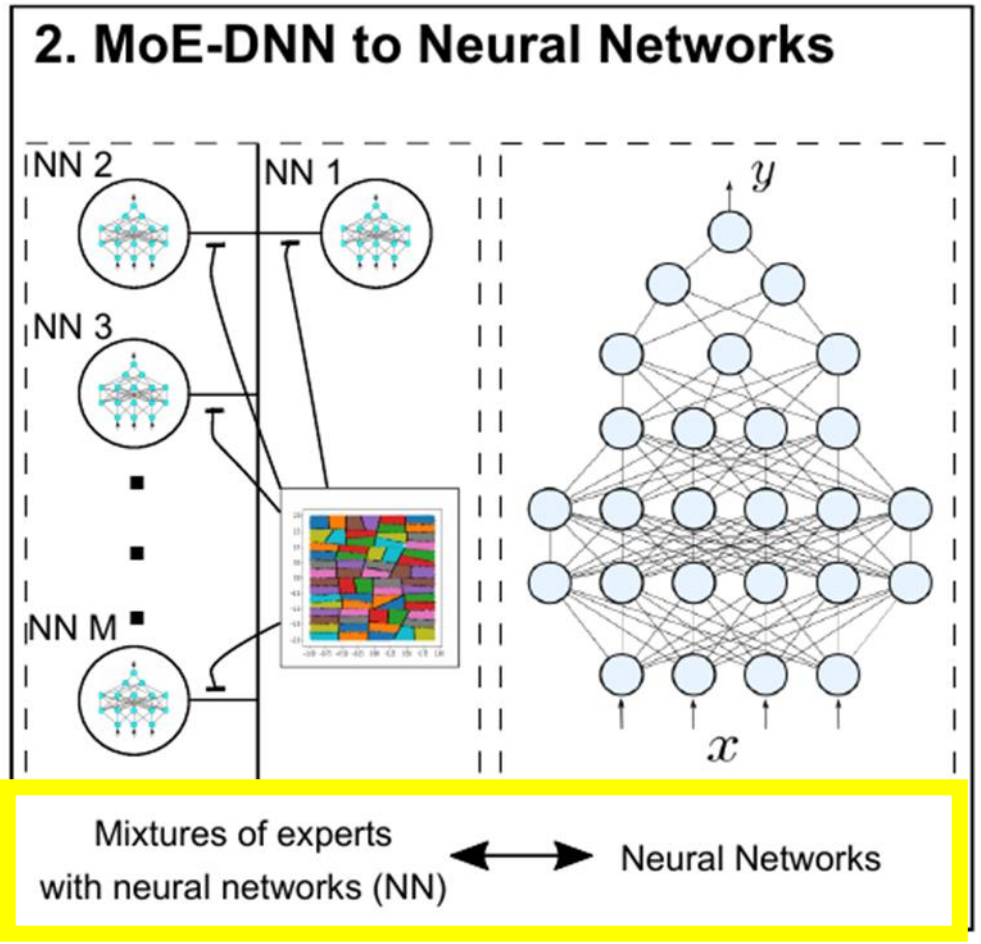
- MoE NN ↔ MoE GP (not the goal)
- A single NN ↔ MoE GP (original goal)

Insight

- Imagine an already trained NN: $f_\theta(x)$
- Assume all NN experts are the same:

  $f_\theta(x) = f_{\theta_1}(x) = f_{\theta_2}(x) \ldots = f_{\theta_M}(x)$
- Due to hard partitioning, input-output relationship are the same:

$$y = \sum_{m=1}^{M} g_m(x) f_\theta(x) = f_\theta(x)$$



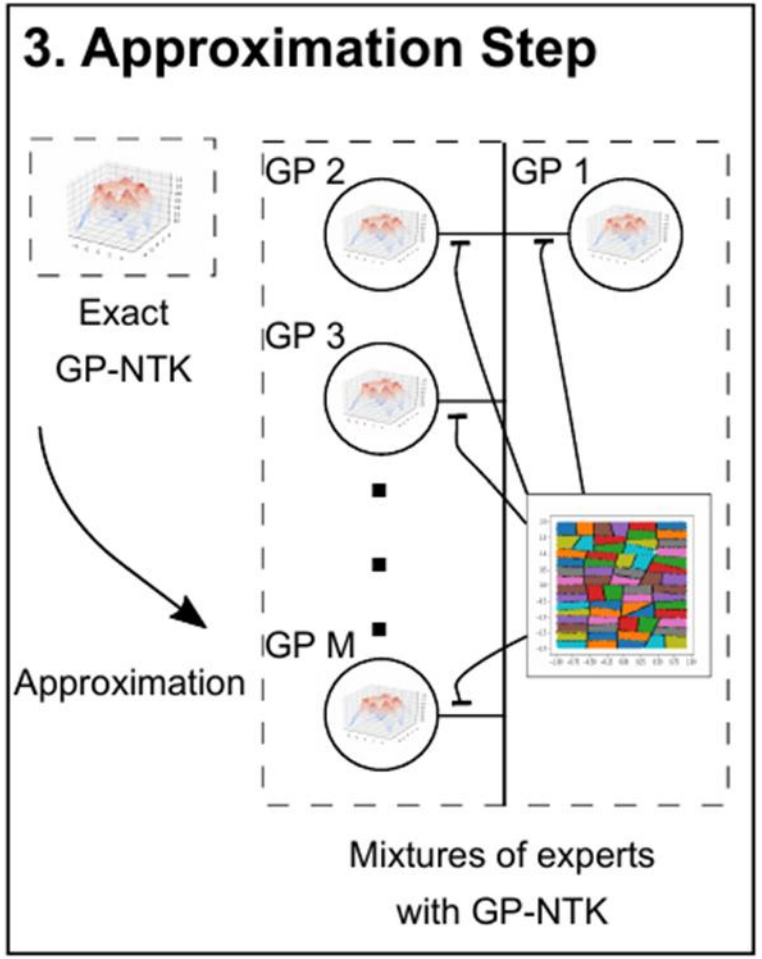
Idea: Theory of Neural Tangent Kernel

Result

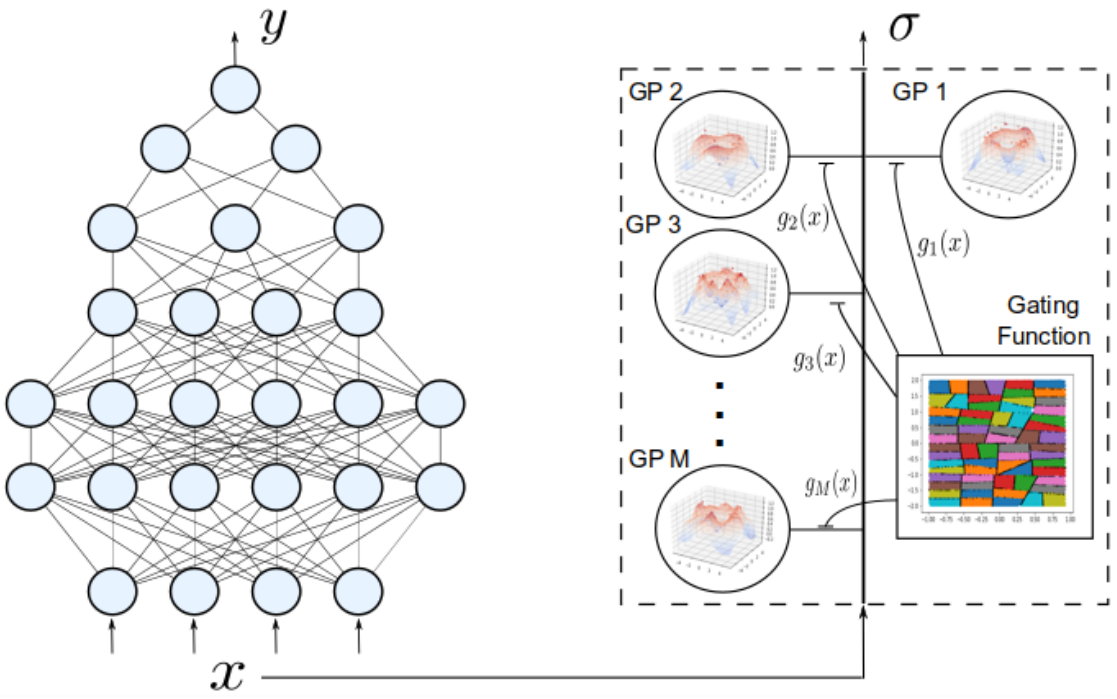
- A single NN imagination → GPs with $\boldsymbol{K = K_1 = K_2 \ldots = K_M}$.
- A full GP ≈ MoE GPs (block diagonal approximation)

Implications

- Neural networks for mean predictions: $\boldsymbol{y = f_w(x)}$
- MoE GP for covariance estimation: $\boldsymbol{\Sigma = K_{**} - K_*^T\left[K + \sigma_n^2 I\right]^{-1} K_*}$
  - The gating function has picked a GP expert.
  - The NTK is a neural network based kernel: $\mathbf{GP}(\mathbf{0}, \frac{1}{\delta_m}\mathbf{J}_{f_m}^{T}(\mathbf{x})\mathbf{J}_{f_m}(\mathbf{x}))$
  - Note: $\mathbf{y} \neq \tilde{\mathbf{y}}$ but $\mathbf{COV}(y) \approx \mathbf{COV}(\tilde{y})$.
- Closed-form solution for uncertainty: $\boldsymbol{p(y^*|x^*, D) = \mathcal{N}\left(f_{\theta=\hat{\theta}(x^*)}, \Sigma\right)}$



Idea: Theory of Neural Tangent Kernel



- Due to the mixtures of experts idea (Jacobs et al 1991), the complexity of full GP reduces (Tresp 2001):

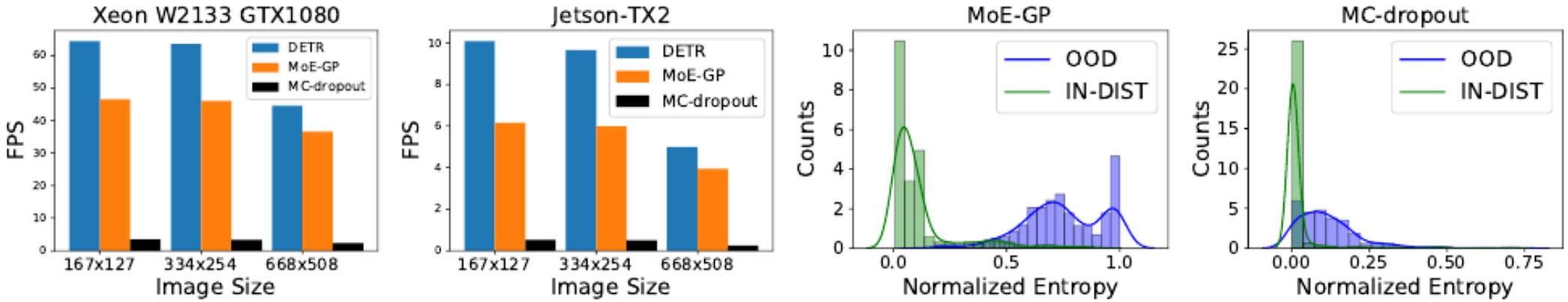
$$\begin{bmatrix} \tilde{\boldsymbol{y}}_{\boldsymbol{m}} \\ \tilde{\boldsymbol{f}}_{\boldsymbol{m}} \end{bmatrix} \sim \boldsymbol{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K_m} + \boldsymbol{\sigma_{0,m}} \boldsymbol{I} & \boldsymbol{k_{m,*}} \\ \boldsymbol{k_{m,*}^T} & \boldsymbol{k_{m,**}} \end{bmatrix}\right)$$

- Analytical method for uncertainty Estimates (“sampling-free”):

$$\boldsymbol{\Sigma_m} = \boldsymbol{k_{m,**}} - \boldsymbol{k_{m,*}^T}(\boldsymbol{K_m} + \boldsymbol{\sigma_0 I})^{-1}\boldsymbol{k_{m,*}} + \boldsymbol{\sigma_0}$$

$$\boldsymbol{p(c|z_m)} = \left(\frac{\boldsymbol{z_m}}{\sqrt{\boldsymbol{1} + \boldsymbol{\lambda_{m,0}}\boldsymbol{\Sigma_m}(\boldsymbol{x^*})}}\right)$$ (Lu et al 2020)

Results: Planetary scenario



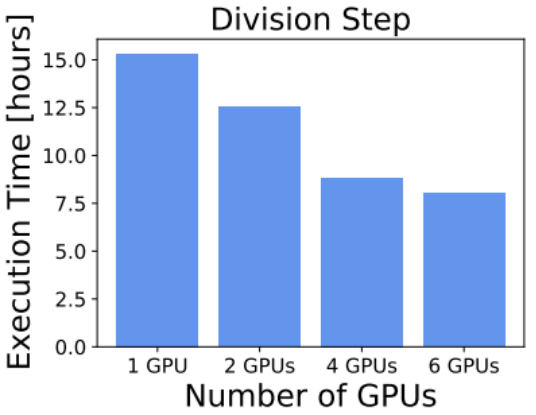
- Run-time comparison on a GPU-desktop and an embedded GPU. Higher FPS the faster.
- Entropy histogram. More separable, better calibrated the uncertainty estimates.

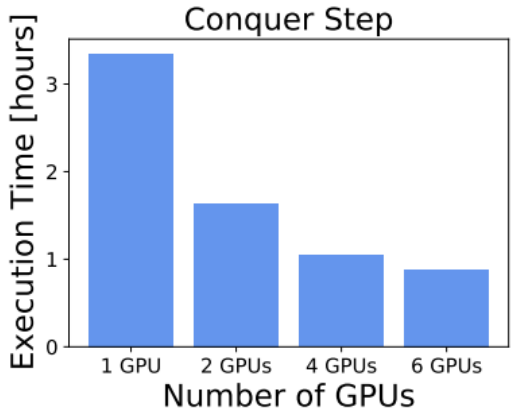
Main take-away/use-cases: when sparse GPs can scale, real-time uncertainty estimates from a GP formulation of neural networks can be obtained, improving over the state-of-the-art methods.

Results: Scalability

- Time to get a GP model for 2 million data points.
- Evaluated on inverse dynamics tasks.

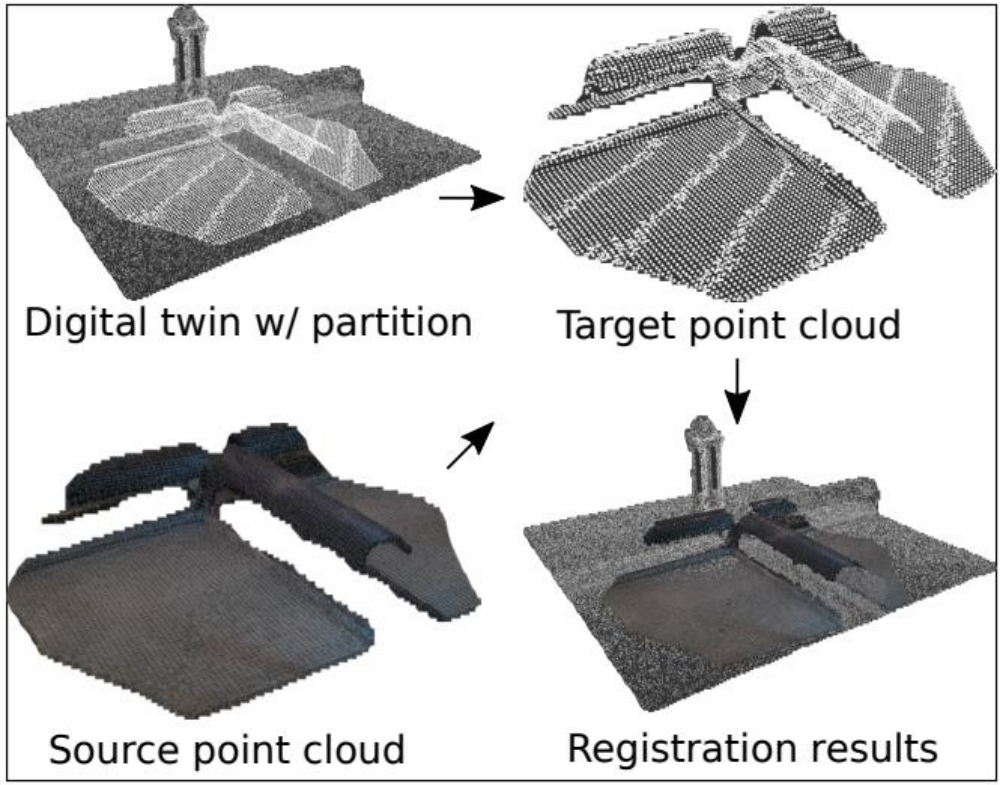
E.g., 180360 train points on kuka1 and kuka2

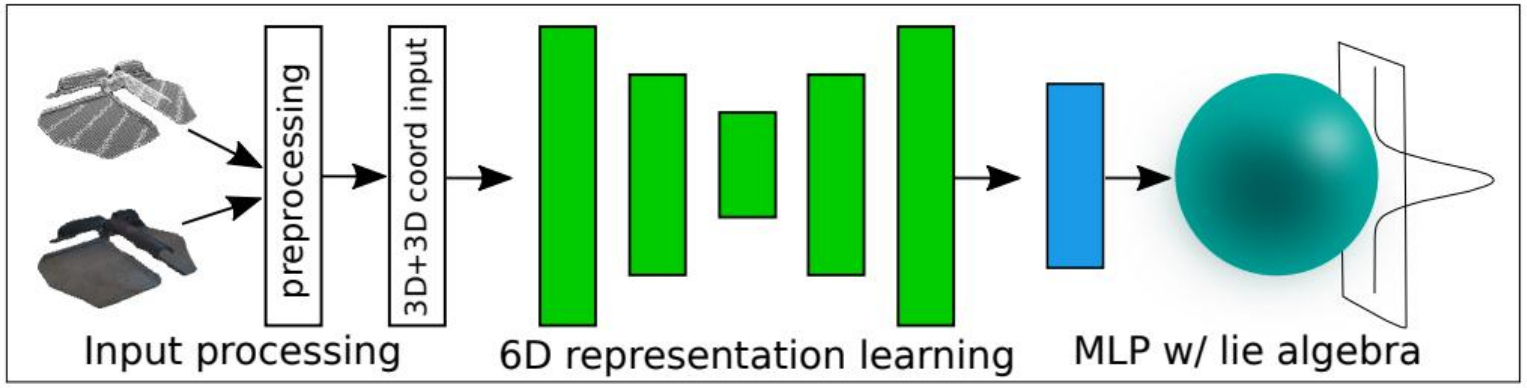


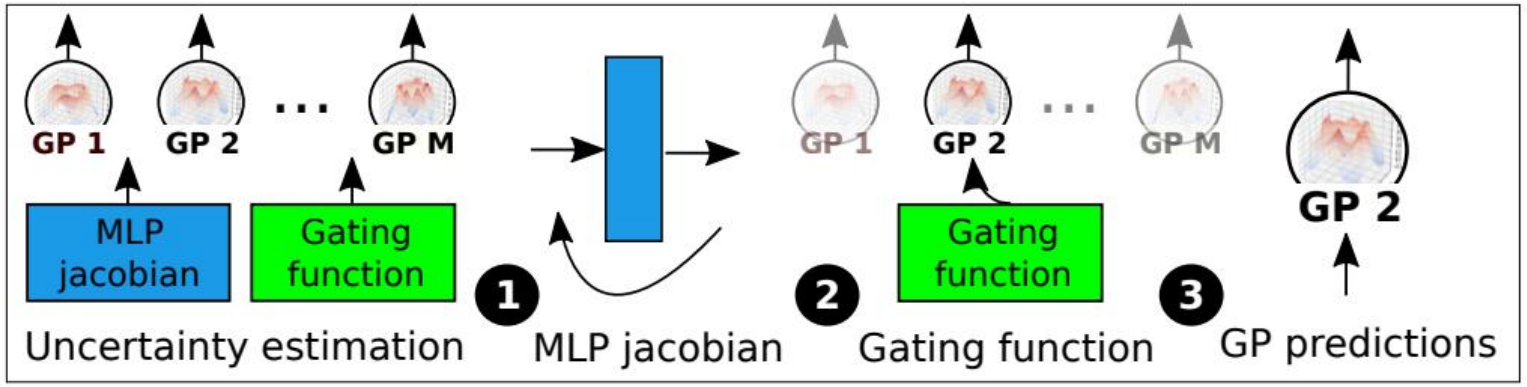


Train	Test	**Ensemble**	**MC-dropout**	**rMC-dropout**	**Laplace**	**rLaplace**	**Ours**
sarcos	sarcos	*1.261±0.105*	1.398±0.012	1.398±0.012	1.392±0.014	1.392±0.014	**1.035±0.039**
sarcos	kuka1	*17.86±2.995*	**16.21±0.866**	20.58±1.144	25.88±1.354	24.55±2.746	21.53±2.268
sarcos	kuka2	*17.50±2.895*	**15.63±0.858**	20.17±0.988	25.59±1.079	24.42±2.257	20.92±2.236
sarcos	kukasim	**23.06±2.649**	*51.09±11.14*	61.40±12.921	77.13±15.40	74.03±13.23	68.95±4.003
kuka1	kuka1	2.013±0.020	*1.611±0.007*	1.620±0.006	1.676±0.013	1.719±0.071	**1.347±0.013**
kuka1	kuka2	2.085±0.005	1.349±0.019	1.330±0.006	**1.310±0.005**	**1.310±0.008**	*1.315±0.003*
kuka1	kukasim	60.44±1.108	*42.14±2.869*	48.76±0.566	60.37±0.799	59.74±1.571	**1.348±0.012**
kuka1	sarcos	*8.128±0.169*	22.24±4.288	42.36±2.398	122.68±13.32	1837±2431.3	**1.356±0.014**
kuka2	kuka2	2.042±0.009	1.443±0.005	*1.423±0.006*	1.429±0.006	*1.423±0.006*	**1.354±0.013**
kuka2	kukasim	63.07±0.741	*38.94±0.561*	48.93±0.779	60.36±1.005	59.05±1.158	**1.333±0.012**
kuka2	sarcos	*8.509±0.461*	18.24±1.599	53.24±6.287	141.7±17.14	211.3±180.2	**1.355±0.014**
kuka2	kuka1	2.106±0.010	1.461±0.004	1.395±0.004	*1.373±0.005*	*1.374±0.004*	**1.331±0.013**

SPIRIT: proposed method

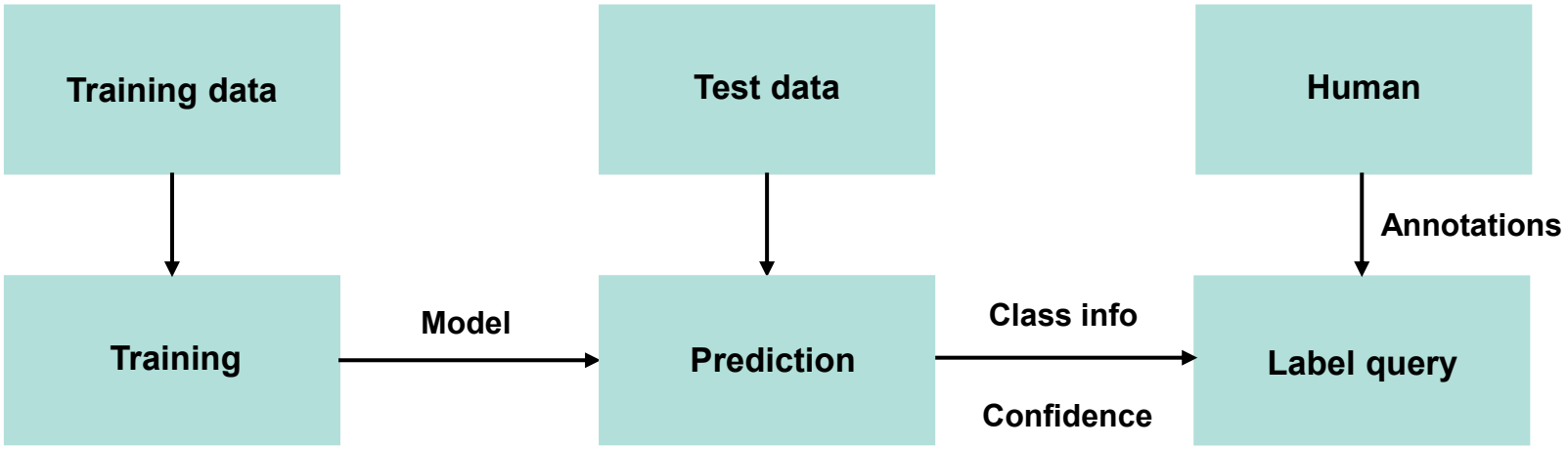


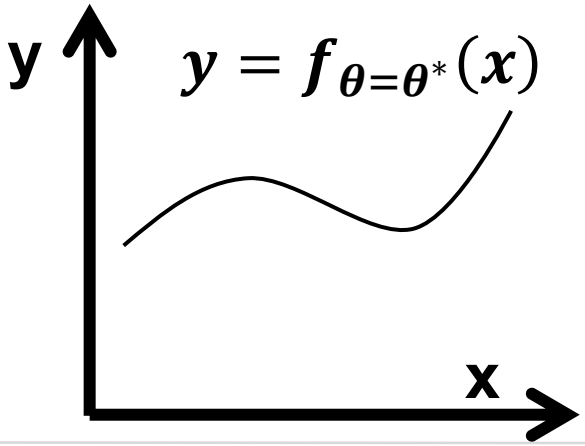


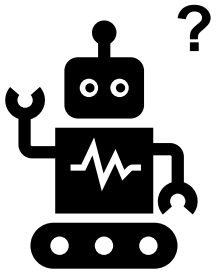


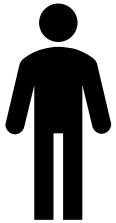
Active Learning

Stream-based active learning





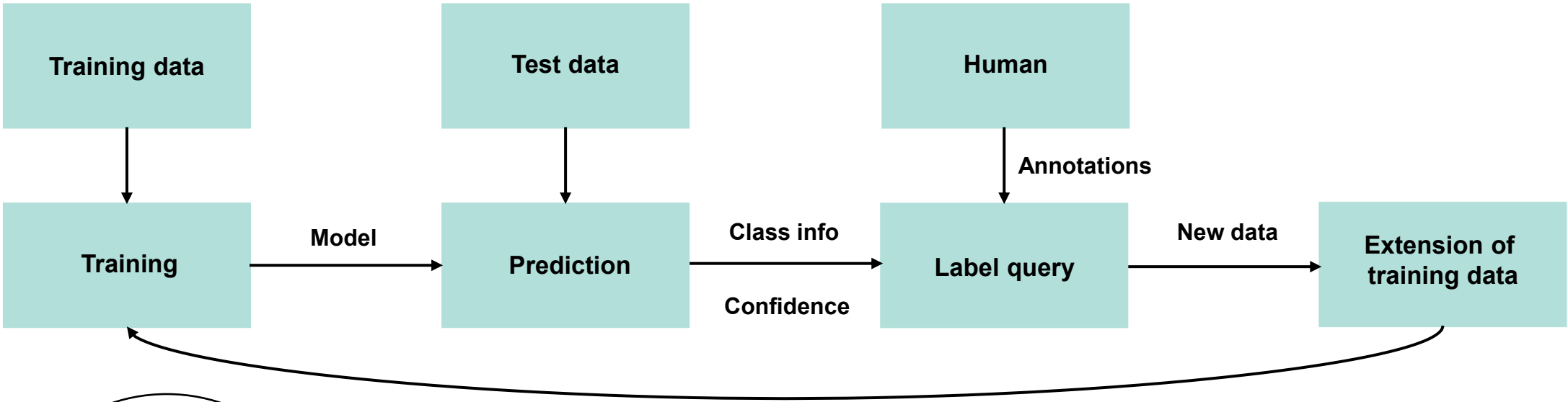


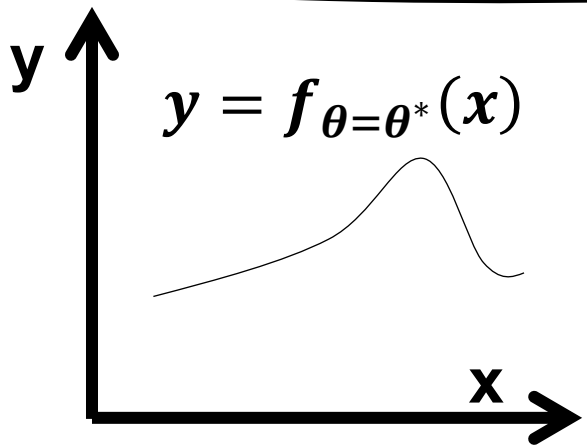


This is a t-shirt!

Lee, J., Birr, T., Triebel, R., & Asfour, T. (2025). CLEVER: Stream-Based Active Learning for Robust Semantic Perception From Human Instructions. *IEEE Robotics and Automation Letters*.

Stream-based active learning





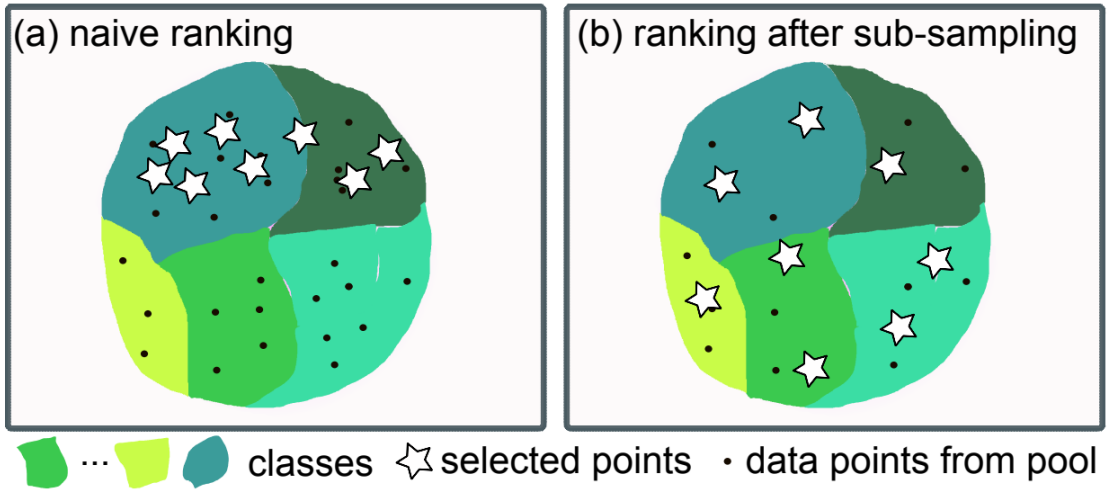
- Learning from limited data.
  - Need good generalization
- Ask for help in a right way.
  - Need good uncertainty.

Lee, J., Birr, T., Triebel, R., & Asfour, T. (2025). CLEVER: Stream-Based Active Learning for Robust Semantic Perception From Human Instructions. *IEEE Robotics and Automation Letters*.

Active learning criteria

$$I[y; \theta \mid x, D] = H[y \mid x, D] - \mathbb{E}_{p(\theta|D)}[H[y \mid x, \theta]]$$

Mutual information:
This measures how much learning the label y at x would reduce uncertainty about the model parameters.



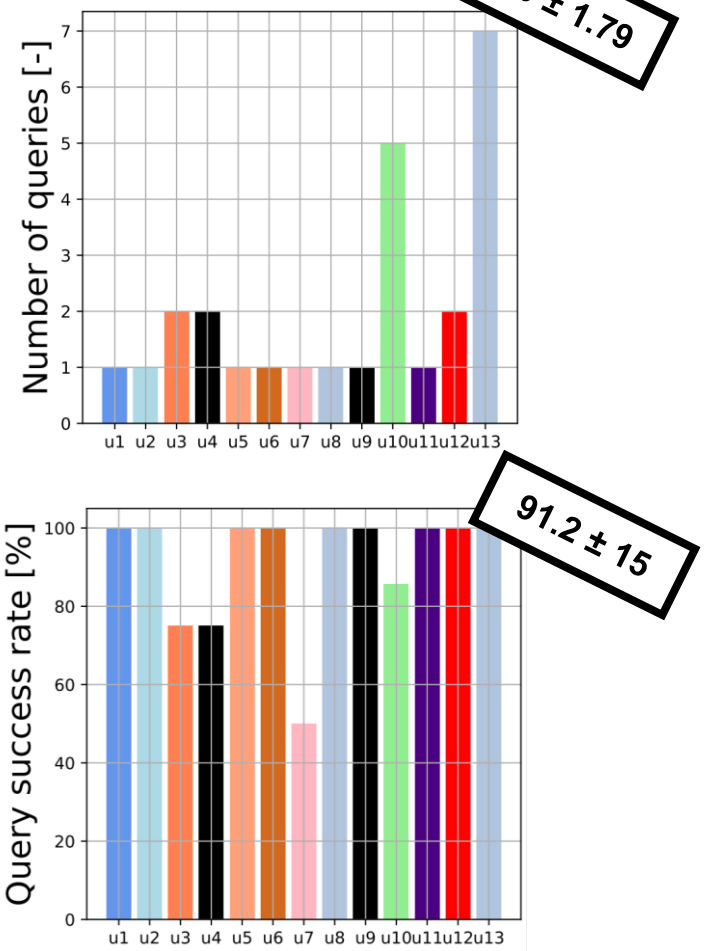
$H[y \mid x, D]$ Predictive entropy — how uncertain the model's predictions are overall.

$\mathbb{E}_{p(\theta|D)}[H[y \mid x, \theta]]$ Expected entropy if we knew which parameter setting θ is true (average over model posterior).

Measures **how much uncertainty comes from the parameters** (epistemic).

Diversity was also important for DL.

Open-set evaluation with users



Lee, J., Birr, T., Triebel, R., & Asfour, T. (2025). CLEVER: Stream-Based Active Learning for Robust Semantic Perception From Human Instructions. *IEEE Robotics and Automation Letters*.