Meta Learning via Bayesian Inference

Marc Deisenroth
UCL Centre for Artificial Intelligence
Department of Computer Science
University College London

@mpd37
m.deisenroth@ucl.ac.uk
https://deisenroth.cc

Creative Machine Learning

Research Seminar @ DeepMind

May 10, 2022

Collaborators

Steindór Sæmundsson

Jean Kaddour

Katja Hofmann

Meta and Transfer Learning

- Motivation: Learn predictive models (and controllers) for different robot arms

Meta and Transfer Learning

- Motivation: Learn predictive models (and controllers) for different robot arms
- Smoothness assumption: Overall the dynamics should not be too dissimilar ⏵⏵⏵ Share some global properties




- Motivation: Learn predictive models (and controllers) for different robot arms
- Smoothness assumption: Overall the dynamics should not be too dissimilar ⏵⏵⏵ Share some global properties
- Sightly different configurations (e.g., mass/link length) ⏵⏵⏵ Differ locally

Meta and Transfer Learning

- Motivation: Learn predictive models (and controllers) for different robot arms
- Smoothness assumption: Overall the dynamics should not be too dissimilar ⏩ Share some global properties
- Sightly different configurations (e.g., mass/link length) ⏩ Differ locally
- Re-use experience gathered so far generalize learning to new dynamics that are similar ⏩ Accelerated learning

Approach

- Consider supervised learning problem for task p: $\boldsymbol{y}^p = f_p(\boldsymbol{x}; \boldsymbol{\theta})$

Approach

- Consider supervised learning problem for task p: $\boldsymbol{y}^p = f_p(\boldsymbol{x}; \boldsymbol{\theta})$
- Introduce local, task-specific latent variable $\boldsymbol{h}_p$, so that

$$\boldsymbol{y}^p = f_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}_p)$$

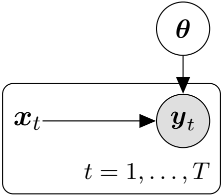
Approach

- Consider supervised learning problem for task p: $\boldsymbol{y}^p = f_p(\boldsymbol{x}; \boldsymbol{\theta})$
- Introduce local, task-specific latent variable $\boldsymbol{h}_p$, so that

$$\boldsymbol{y}^p = f_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}_p)$$

- Separate global from local (task-specific) properties
- Shared global parameters θ describe general "shape" of the function/dynamics
- Task-specific properties described by latent variable $\boldsymbol{h}$

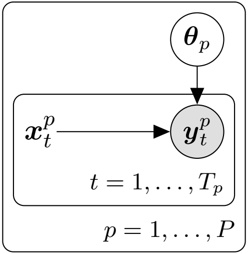
Model



- Single-task supervised learning

Sæmundsson et al. (UAI, 2018): Meta Reinforcement Learning with Latent Variable Gaussian Processes

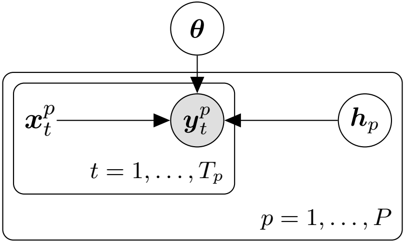
Model



- Multi-task supervised learning (independence between tasks)

Sæmundsson et al. (UAI, 2018): Meta Reinforcement Learning with Latent Variable Gaussian Processes

Model

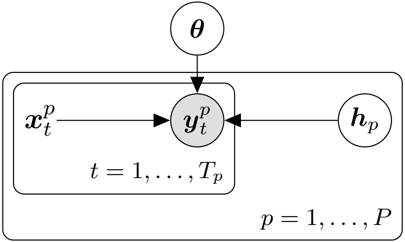


- Meta learning setting (see also Gordon et al. (2019) for a similar setting):

$$\boldsymbol{y}_t^p = f_{\boldsymbol{\theta}}(\boldsymbol{x}_t, \boldsymbol{h}_p)$$

  - Parameters θ capture global properties of the model
  - Latent variable $\boldsymbol{h}_p$ describes local configuration
  - Share (global) properties between tasks

Sæmundsson et al. (UAI, 2018): Meta Reinforcement Learning with Latent Variable Gaussian Processes



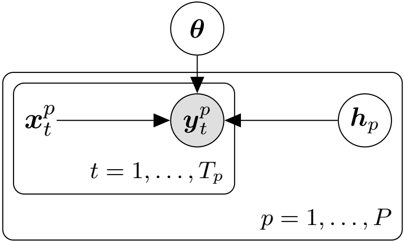
$$\boldsymbol{y}_t^p = f_{\boldsymbol{\theta}}(\boldsymbol{x}_t, \boldsymbol{h}_p) + \boldsymbol{\epsilon}$$

$\boldsymbol{f}_{\boldsymbol{\theta}}(\cdot) \sim GP$ ⏩ SV-GP (Titsias, 2009)

$q(\boldsymbol{H}) = \prod_{p=1}^{P} \mathcal{N}(\boldsymbol{h}_p | \boldsymbol{n}_p, \boldsymbol{T}_p)$

Sæmundsson et al. (UAI, 2018): Meta Reinforcement Learning with Latent Variable Gaussian Processes

Model: Some Specifics



$$\boldsymbol{y}_t^p = f_{\boldsymbol{\theta}}(\boldsymbol{x}_t, \boldsymbol{h}_p) + \boldsymbol{\epsilon}$$

$\boldsymbol{f}_{\boldsymbol{\theta}}(\cdot) \sim GP$ ⏩ SV-GP (Titsias, 2009)

$$q(\boldsymbol{H}) = \prod_{p=1}^{P} \mathcal{N}(\boldsymbol{h}_p | \boldsymbol{n}_p, \boldsymbol{T}_p)$$

$$p(\boldsymbol{Y}, \boldsymbol{H}, \boldsymbol{f}(\cdot) | \boldsymbol{X}) = \prod_{p=1}^{P} q(\boldsymbol{h}_p) \prod_{t=1}^{T_p} p(\boldsymbol{y}_t | \boldsymbol{x}_t, \boldsymbol{h}_p, \boldsymbol{f}(\cdot)) q(\boldsymbol{f}(\cdot))$$

Sæmundsson et al. (UAI, 2018): Meta Reinforcement Learning with Latent Variable Gaussian Processes

Training and Predictions

- Training data
  - $(\boldsymbol{x}_t^p, \boldsymbol{y}_t^p)$ for $t = 1, \dots, T_p$ for $p = 1, \dots, P$ tasks
  - Assume that the task identity at training time is known
  - Learn global model parameters $\boldsymbol{\theta}$ and variational parameters of $q(\boldsymbol{h}_1, \dots, \boldsymbol{h}_P)$.

Training and Predictions

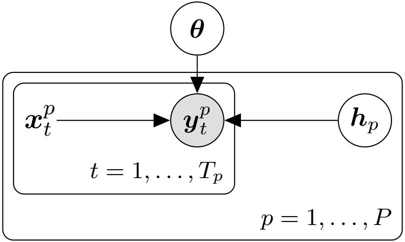
- Training data
  - $(\boldsymbol{x}_t^p, \boldsymbol{y}_t^p)$ for $t = 1, \dots, T_p$ for $p = 1, \dots, P$ tasks
  - Assume that the task identity at training time is known
  - Learn global model parameters $\boldsymbol{\theta}$ and variational parameters of $q(\boldsymbol{h}_1, \dots, \boldsymbol{h}_P)$.
- Test data
  - $(\boldsymbol{x}_t^i, \boldsymbol{y}_t^i)$ for some t ▶▶▶ Posterior on $\boldsymbol{h}_i$
  - $\boldsymbol{x}_t^i$ for some t ▶▶▶ Predict $\boldsymbol{y}_t^i$ using prior/posterior on $\boldsymbol{h}_i$

Training and Predictions

- Training data
  - $(\boldsymbol{x}_t^p, \boldsymbol{y}_t^p)$ for $t = 1, \dots, T_p$ for $p = 1, \dots, P$ tasks
  - Assume that the task identity at training time is known
  - Learn global model parameters $\boldsymbol{\theta}$ and variational parameters of $q(\boldsymbol{h}_1, \dots, \boldsymbol{h}_P)$.
- Test data
  - $(\boldsymbol{x}_t^i, \boldsymbol{y}_t^i)$ for some t ⏩ Posterior on $\boldsymbol{h}_i$
  - $\boldsymbol{x}_t^i$ for some t ⏩ Predict $\boldsymbol{y}_t^i$ using prior/posterior on $\boldsymbol{h}_i$
- Zero/few-shot predictions at new tasks

$$p(\boldsymbol{y}_*|\boldsymbol{x}_*) = \int p(\boldsymbol{y}_*|\boldsymbol{x}_*, \boldsymbol{h}_*)q(\boldsymbol{h}_*)d\boldsymbol{h}_*$$

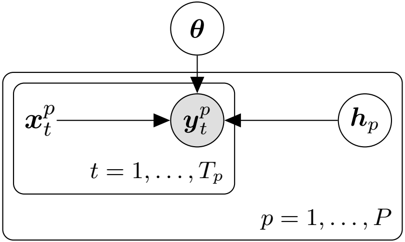
Training



- Mean-field variational family:

$$q(\boldsymbol{f}(\cdot), \boldsymbol{H}) = q(\boldsymbol{f}(\cdot))q(\boldsymbol{H})$$

Training



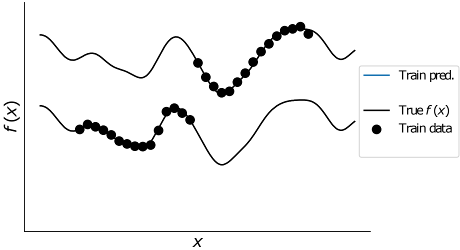
- Mean-field variational family:

$$q(\boldsymbol{f}(\cdot), \boldsymbol{H}) = q(\boldsymbol{f}(\cdot))q(\boldsymbol{H})$$

- Maximize lower bound on the model evidence (ELBO):

$$ELBO = \mathbb{E}_{q(\boldsymbol{f}(\cdot),\boldsymbol{H})}\left[\log \frac{p(\boldsymbol{Y}, \boldsymbol{H}, \boldsymbol{f}(\cdot)|\boldsymbol{X})}{q(\boldsymbol{f}(\cdot), \boldsymbol{H})}\right]$$

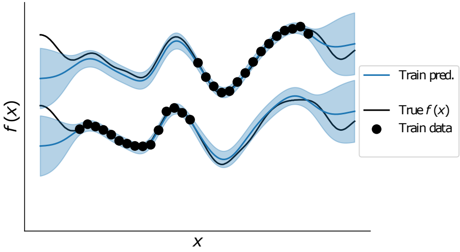
Example



$$y = f(x) + h_p + \epsilon$$

- Training data (black discs) from 2 training tasks

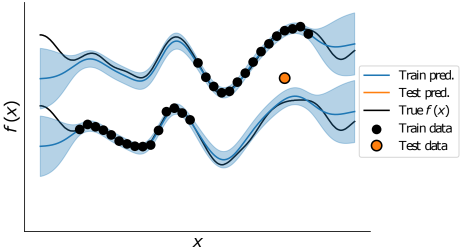
Example



$$y = f(x) + h_p + \epsilon$$

- Training data (black discs) from 2 training tasks
- Gaussian process models "shape" of the function
- Latent variable h_p describes vertical offset
- Good generalization at training tasks due to latent variable

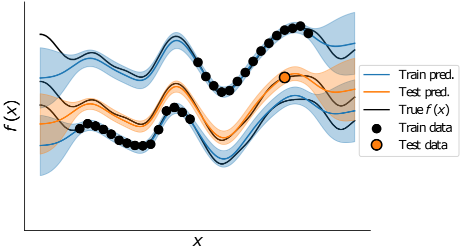
Example



$$y = f(x) + h_p + \epsilon$$

- Training data (black discs) from 2 training tasks
- Gaussian process models "shape" of the function
- Latent variable h_p describes vertical offset
- Good generalization at training tasks due to latent variable
- 1-shot predictions at new tasks are informed (orange)

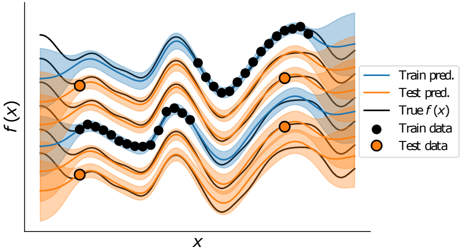
Example



$$y = f(x) + h_p + \epsilon$$

- Training data (black discs) from 2 training tasks
- Gaussian process models "shape" of the function
- Latent variable h_p describes vertical offset
- Good generalization at training tasks due to latent variable
- 1-shot predictions at new tasks are informed (orange)

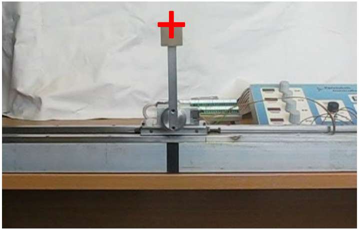
Example



$$y = f(x) + h_p + \epsilon$$

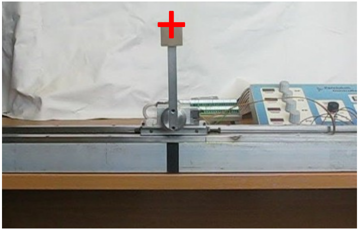
- Training data (black discs) from 2 training tasks
- Gaussian process models "shape" of the function
- Latent variable h_p describes vertical offset
- Good generalization at training tasks due to latent variable
- 1-shot predictions at new tasks are informed (orange)

Model-based RL: Cart-Pole Swing-up



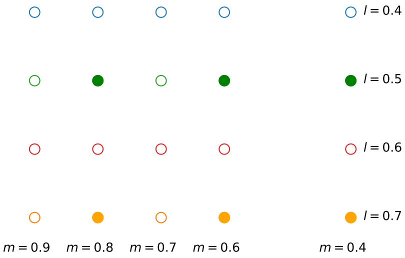
- Learn dynamics and controllers for different cart-pole systems (lengths and masses of pendulum change)

Model-based RL: Cart-Pole Swing-up



- Learn dynamics and controllers for different cart-pole systems (lengths and masses of pendulum change)
- Model-based RL algorithm (Kamthe & Deisenroth, 2018)
  - Gaussian process as learned dynamics model
  - Moment matching for long-term planning
  - Model predictive control for policy learning

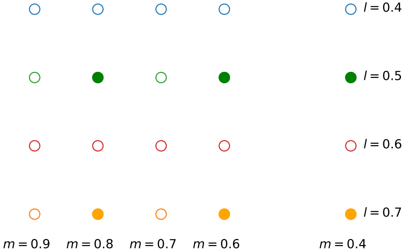
Latent Embeddings

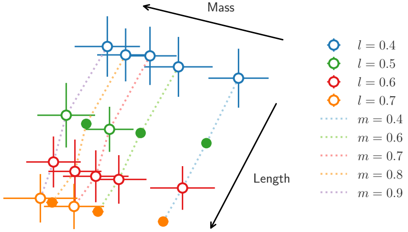


- Latent variable $\boldsymbol{h}$ encodes length l and mass m of the pole
- 6 training tasks, 14 held-out test tasks
- Left: True configurations;

Sæmundsson et al. (UAI, 2018): Meta Reinforcement Learning with Latent Variable Gaussian Processes

Latent Embeddings

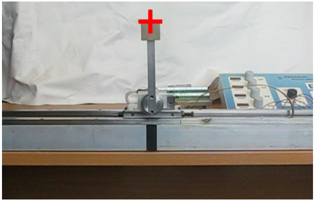




- Latent variable $\boldsymbol{h}$ encodes length l and mass m of the pole
- 6 training tasks, 14 held-out test tasks
- Left: True configurations; Right: learned embeddings

Sæmundsson et al. (UAI, 2018): Meta Reinforcement Learning with Latent Variable Gaussian Processes

Meta-RL (Cart Pole): Training

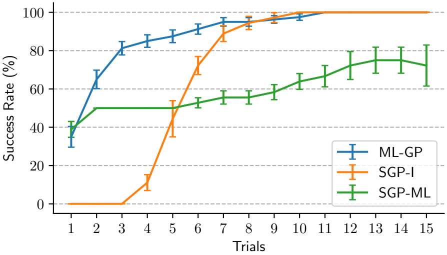


- Pre-trained on 6 training configurations until solved

Model	Training (s)	Description
Independent	16.1 ± 0.4	Independent GP-MPC
Aggregated	23.7 ± 1.4	Aggregated experience (no latents)
Meta learning	15.1 ± 0.5	Aggregated experience (with latents)

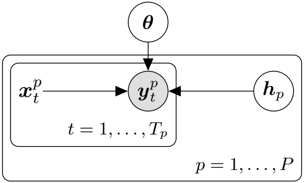
▸▸▸ Meta learning can speed up multi-task RL

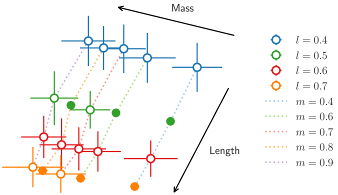
Meta-RL (Cart Pole): Few-Shot Generalization



- Few-shot generalization on 4 unseen configurations
- Success: solve all 10 (6 training + 4 test) tasks
- Meta learning
- Independent (GP-MPC)
- Aggregated experience model (no latents)

▶▶▶ Meta RL generalizes well to unseen tasks





- Formulate meta learning as a hierarchical Bayesian inference problem
- Automatically infer similarities between tasks via latent variables
- Speed up multi-task (reinforcement) learning
- Few-shot learning of new tasks

Active Meta Learning: Setting




- Training tasks are not given a priori

Active Meta Learning: Setting




- Training tasks are not given a priori
- "What task to learn next?"

Active Meta Learning: Setting



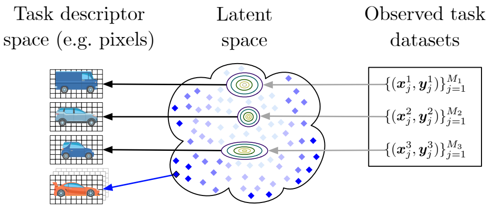

- Training tasks are not given a priori
- "What task to learn next?"
- Objective: Given a space of admissible tasks, choose a (small) set of tasks that allow us to "cover" the entire task space

Active Meta Learning: Setting




- Training tasks are not given a priori
- “What task to learn next?”
- Objective: Given a space of admissible tasks, choose a (small) set of tasks that allow us to “cover” the entire task space
- Idea: use probabilistic latent embeddings of tasks for efficient exploration (active learning)

Active Meta Learning Setting

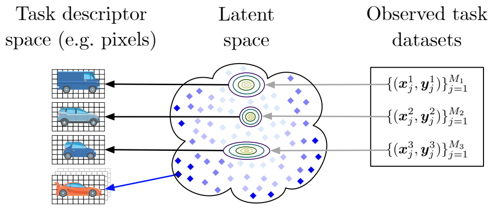


Approach:

- Observe task descriptors: e.g. task parametrizations, tactile information, pixel observations
- Probabilistic latent embedding of task (descriptors)

Kaddour et al. (NeurIPS, 2020): Probabilistic Active Meta-Learning

Active Meta Learning Setting

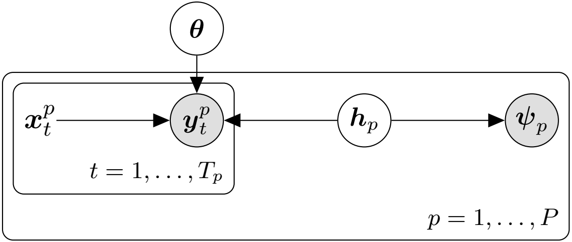


Approach:

- Observe task descriptors: e.g. task parametrizations, tactile information, pixel observations
- Probabilistic latent embedding of task (descriptors)
- Specify a discrete set of task descriptors, infer their latent embedding
- Define a “surprise” utility function in latent space and find “best” candidate

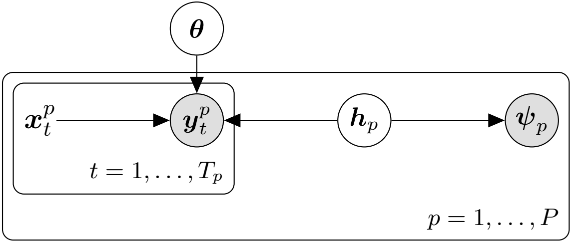
Kaddour et al. (NeurIPS, 2020): Probabilistic Active Meta-Learning

Model



- Task descriptors ψ_p (e.g., physical properties, images, ...) *as* additional observations (of the task)

Model



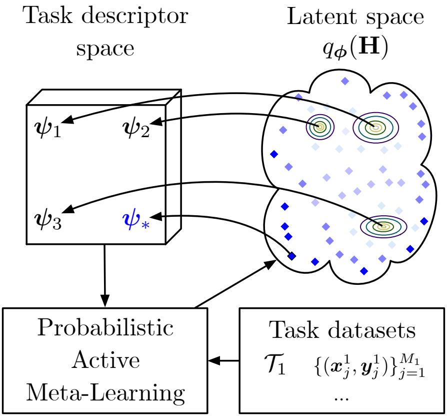
- Task descriptors $\boldsymbol{\psi}_p$ (e.g., physical properties, images, ...) as additional observations (of the task)
- ELBO

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{Y}, \boldsymbol{\Psi}|\boldsymbol{X}) = \log \mathbb{E}_{q_\phi(\boldsymbol{H})}\left[p_{\boldsymbol{\theta}}(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{H}) p_{\boldsymbol{\theta}}(\boldsymbol{\Psi}|\boldsymbol{H}) \frac{p(\boldsymbol{H})}{q_\phi(\boldsymbol{H})}\right]$$

$$\leqslant \mathcal{L}_{ML} + \sum_{p=1}^{P} \mathbb{E}_{q_{\phi_p}(\boldsymbol{h}_p)}[\log p_{\boldsymbol{\theta}}(\boldsymbol{\psi}_p|\boldsymbol{h}_p)]$$

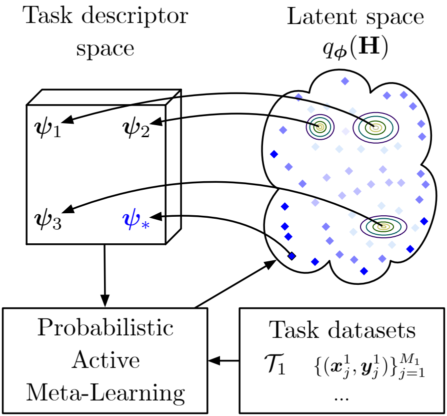
Kaddour et al. (NeurIPS, 2020): Probabilistic Active Meta-Learning

Exploration in Latent Space



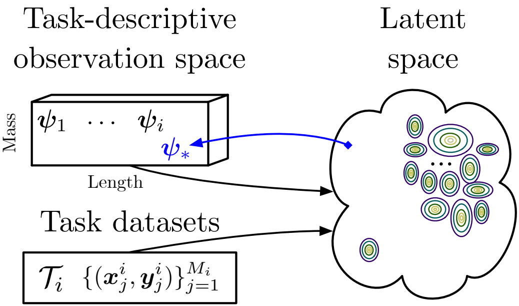
- Exploration in the latent space
  ▸▸▸ Exploit learned similarities between tasks

Exploration in Latent Space



- Exploration in the latent space
  ⏩ Exploit learned similarities between tasks
- Latent space characterized by Gaussian mixture distribution (variational posteriors of previous tasks)

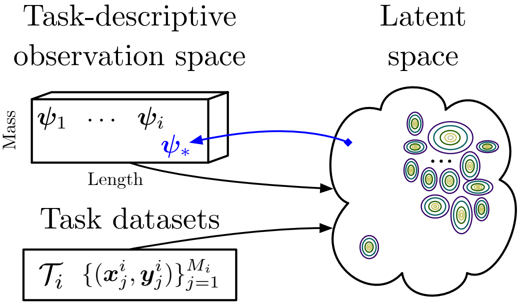
Utility Function and Exploration



- Utility: Negative log-likelihood of the GMM given test task $\boldsymbol{h}_*$

$$u(\boldsymbol{h}_*) = -\log \sum_{p=1}^{P} q_{\phi_p}(\boldsymbol{h}_*)$$

Utility Function and Exploration

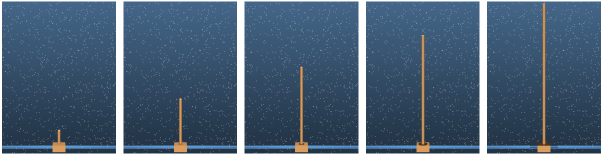


- Utility: Negative log-likelihood of the GMM given test task $\boldsymbol{h}_*$

$$u(\boldsymbol{h}_*) = -\log \sum_{p=1}^{P} q_{\phi_p}(\boldsymbol{h}_*)$$

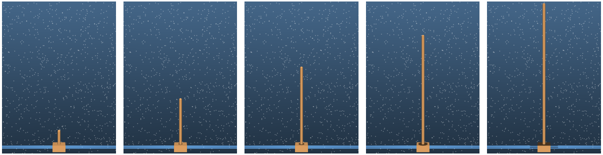
- Rank set of candidate tasks and choose the one with the highest utility

Experiments



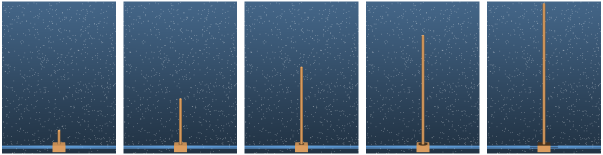
- Objective: Learn good forward models for a range of cart-pole tasks

Experiments



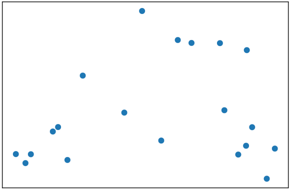
- Objective: Learn good forward models for a range of cart-pole tasks
- Continuous task space defined by varying masses of cart/pole and length of pole

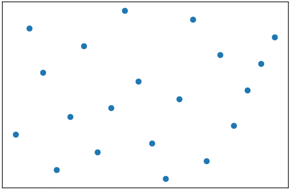
Experiments



- Objective: Learn good forward models for a range of cart-pole tasks
- Continuous task space defined by varying masses of cart/pole and length of pole
- Initialize with 4 tasks; Add 15 more by using different task-sampling strategies
- Evaluate performance on a dense grid of test tasks (NLL and RMSE)

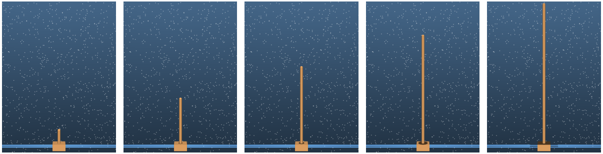
Task Sampling Strategies





- Uniform sampling (UNI)
- Latin hypercube sampling (LHS)
- PAML (probabilistic active meta learning)

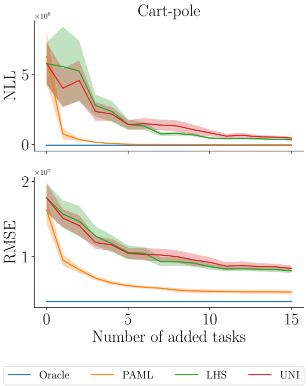
Task Descriptors



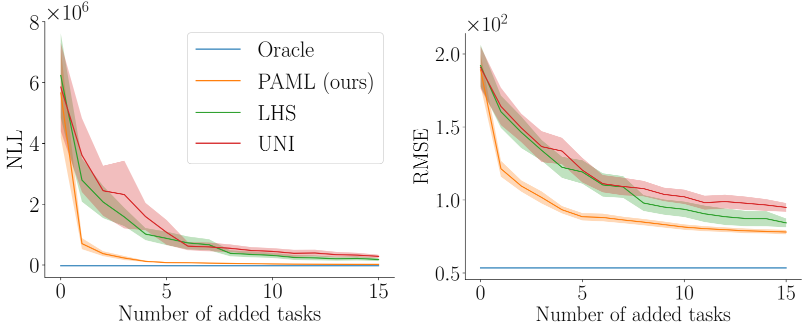
- Exact observations of the task parameters
- Partial observations (only observe changes in length, but not in mass)
- High-dimensional task descriptors (pixels)

Exact Task Descriptors

- Latent embedding h_p learned via variational inference
- PAML approach significantly more efficient in covering all admissible tasks than other sampling approaches

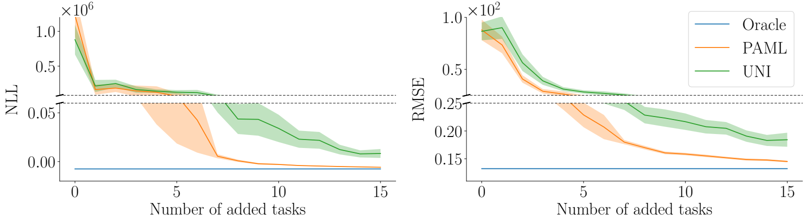


Partial Task Descriptors



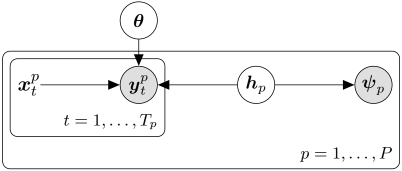
- Only observe change in length, but not in mass
- Overall the same picture as before
- Some loss in learning speed and overall quality of the solution

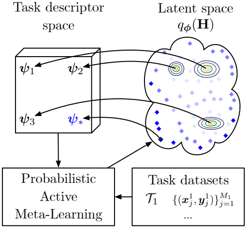
Pixel Task Descriptors



- Task descriptor is a single image of 100 tasks in their initial state (upright pole)
- Pole length varies between $[0.5, 4.5]$m
- VAE for latent embedding
  - Additional reconstruction loss in training objective
- Finds good solution to all tasks quickly

Summary





- Meta learning as a hierarchical Bayesian inference problem
- Learn latent task representation that characterizes task similarities
- Active learning approach in latent space for active task selection
- Code: `https://github.com/JeanKaddour/PAML`

References I

[1] J. Gordon, J. Bronskill, M. Bauer, S. Nowozin, and R. Turner. Meta-Learning Probabilistic Inference for Prediction. In Proceedings of the International Conference on Learning Representations, 2019.

[2] J. Kaddour, S. Sæmundsson, and M. P. Deisenroth. Probabilistic Active Meta-Learning. In Advances in Neural Information Processing Systems, 2020.

[3] S. Kamthe and M. P. Deisenroth. Data-Efficient Reinforcement Learning with Probabilistic Model Predictive Control. In Proceedings of the International Conference on Artificial Intelligence and Statistics, 2018.

[4] S. Sæmundsson, K. Hofmann, and M. P. Deisenroth. Meta Reinforcement Learning with Latent Variable Gaussian Processes. In Proceedings of the Conference on Uncertainty in Artificial Intelligence, 2018.

[5] M. K. Titsias. Variational Learning of Inducing Variables in Sparse Gaussian Processes. In Proceedings of the International Conference on Artificial Intelligence and Statistics, 2009.