Meta Reinforcement Learning as Task Inference

Jan Humplik et al., Google DeepMind

Changhoon, Kevin Jeong

Seoul National University
chjeong@bi.snu.ac.kr

April 6, 2020

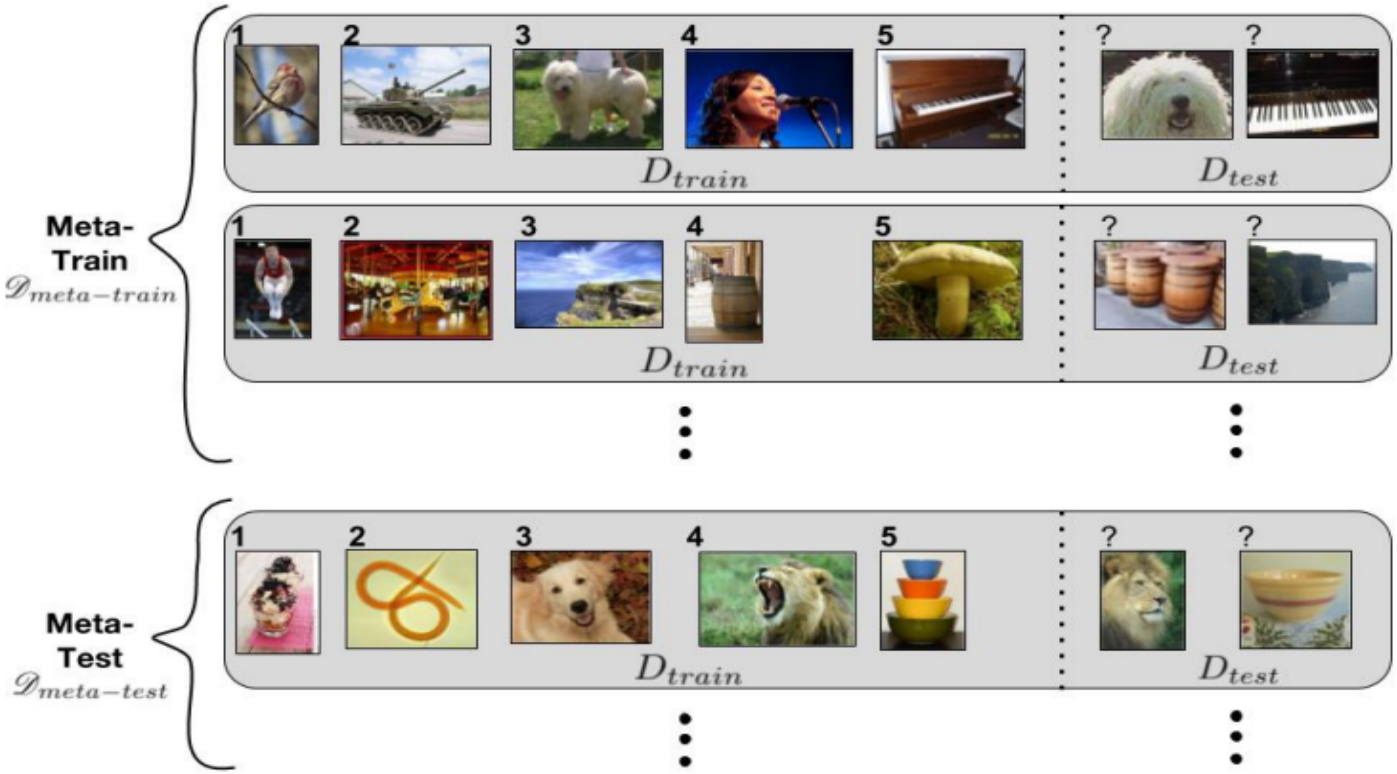
Contents

- Meta Reinforcement Learning
- Preliminaries : POMDP(Partially-Observable Markov Decision Processes)
- Meta Reinforcement Learning and Task Inference
- Experiments
- Conclusion

Meta Learning as Supervised Learning

- Meta Learning(Computer Science)
  - Meta learning is a subfield of machine learning where automatic learning algorithms are applied on metadata about machine learning experiments : ***learning to learn***

- How to learn the optimal $p(\phi_i|D_i^{\text{tr}}, \theta)$ using meta-dataset?



Meta Learning as Supervised Learning

- Black-Box adaptation
  - Train a neural network to represent $p(\phi_i | D_i^{\text{tr}}, \theta)$
  - Use deterministic (point estimate) $\phi_i = f_\theta(D_i^{\text{tr}})$
  - MANN, SNAIL, Meta-Nets, etc.
- Optimization-based Inference
  - Acquire through optimization
  - Meta-parameters θ serve as a prior, so what form of prior?
  - One successful form of prior knowledge: initialization for fine-tuning, MAML, Reptile, etc.
- Non-Parametric methods
  - Use non-parametric learner
  - Siamese nets, Matching nets, ProtoNets, etc.
- etc.

Meta Reinforcement Learning

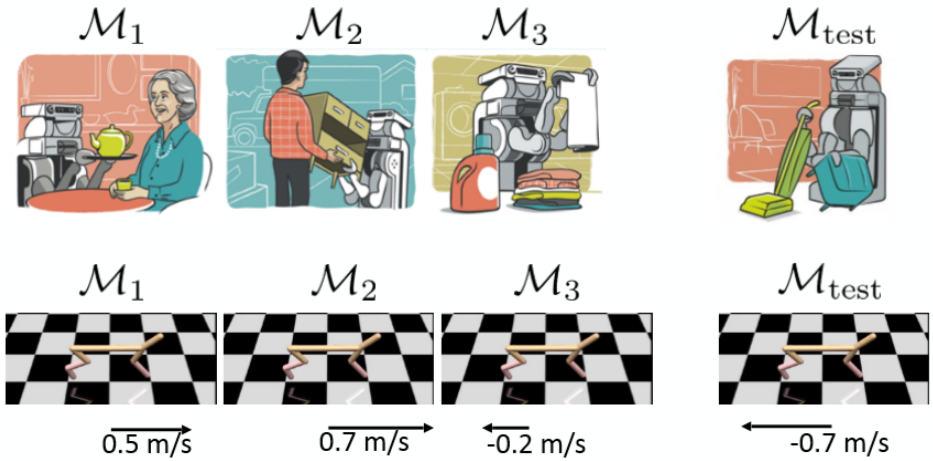
- Can we meta-learn reinforcement learning "algorithms" that are much more efficient?
  - Reinforcement Learning

    $\theta^* = \text{argmax}_\theta \mathbb{E}_{\pi_\theta(\tau)}[R(\tau)]$

    $= f_{\text{RL}}(\mathcal{M})$, where $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, r\}$
  - Meta Reinforcement Learning

    $\theta^* = \text{argmax}_\theta \sum_{i=1}^{n} \mathbb{E}_{\pi_{\phi_i}(\tau)}[R(\tau)]$

    where, $\phi_i = f_\theta(M_i)$



Meta Reinforcement Learning

- Meta-RL with recurrent policies
  - Just RNN it to solve $p(\phi_i|D_i^{\text{tr}}, \theta)$
  - Similar to Black-Box adaption as Meta-Supervised Learning
- Meta-RL as an optimization problem
  - bi-level optimization, MAML for RL
- Meta-RL as partially observed RL
  - it's one of inference problem
  - $\pi_\theta(a|s, z)$, $z_t \sim p(z_t|s_{1:t}, a_{1:t}, r_{1:t})$
  - $z \rightarrow$ everything needed to solve the task
- etc.

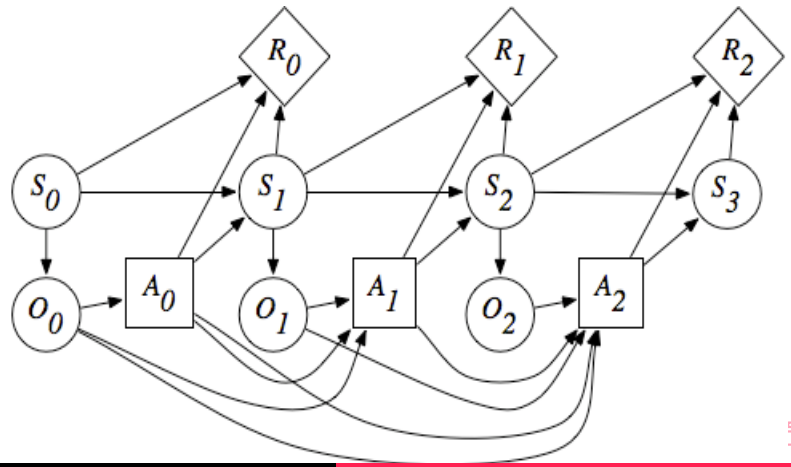
Preliminaries: POMDP(Partially-Observable Markov Decision Processes)

- MDPs(Markov Decision Processes) : $(\mathcal{X}, \mathcal{A}, P, p_0, R, \gamma)$
  - $\mathcal{X}$: the state space
  - $\mathcal{A}$: the action space
  - $P(x'|x, a)$: the transition probability
  - $p_0(x)$: the initial state distribution
  - $R(r|x, a, x')$: the probability of obtaining reward r
  - γ : the discount factor

Markov Model		**Do you have control over the state transition?**	
		No	**Yes**
Are the states completely observable?	**Yes**	Markov Chain	MDP
	No	HMM	POMDP

Preliminaries: POMDP(Partially-Observable Markov Decision Processes)

- POMDP(Partially-Observable Markov Decision Processes) : $(\mathcal{X}, \mathcal{A}, P, p_0, R, \Omega, O, \gamma)$
  - $\mathcal{X}$: the state space
  - $\mathcal{A}$: the action space
  - $P(x'|x, a)$: the transition probability
  - $p_0(x)$: the initial state distribution
  - $R(r|x, a, x')$: the probability of obtaining reward r
  - Ω : the observation state
  - $O(o'|x', a)$: the probability of observing $o' \in \Omega$
  - γ : the discount factor



Preliminaries: POMDP(Partially-Observable Markov Decision Processes)

- The solution to a POMDP
  - The observed trajectory as $\tau_{0:t} = (o_{0:t}, a_{0:t-1}, r_{0:t-1})$
  - Find a policy $\pi^*(a_t|\tau_{0:t})$ which maximizes discounted return i.e. $\pi^* = \text{argmax}_\pi \mathbb{E}_{\tau_{0:\infty}, x_{0:\infty} \sim p_\pi} [\sum_{t=0}^{\infty} \gamma^t r_t]$

- Belief state
  - The optimal policy's dependence on the trajectory can be summarized using *belief* state $b_t(x) \equiv p_\pi(x_t = x|\tau_{0:t})$
  - The *belief* state is a sufficient statistic for optimal a_t in the sense that $\pi^*(a_t|\tau_{0:t}) = \pi^*(a_t|b_t)$

Meta Reinforcement Learning and Task Inference

- Assumptions for the POMDPs
  - $\mathcal{W}$: the space of tasks
  - $p(w)$: the distribution over tasks $w \in \mathcal{W}$
  - each task w is a MDP $(\mathcal{X}, \mathcal{A}, P^w, p_0^w, R^w, \gamma)$
  - train an agent that maximizes future discounted rewards based on a 'small' number of interactions with unknown task w drawn from $p(w)$
  - A single episode per one MDP; multi-episode interactions can be achieved by changing the MDPs(reset init.)

Meta Reinforcement Learning and Task Inference

- Formulation of meta-RL problem using POMDPs
  - sharing the same $\mathcal{A}$ as MDPs
  - states $(x, w) \in \mathcal{X} \times \mathcal{W}$
  - transtion distribution $P(x', w'|x, w, a) = \delta(w' - w)P^w(x'|x, a)$
  - initial distribution $p_0(x, w) = p(w)p_0(x|w)$
  - reward distribution $R(r|x, w, a, x', w') = R^w(r|x, a, x')$
  - deterministic observations $O(o'|x', w, a) = \delta(o' - x')$

- Assumption of the POMDPs: An agent
  - The optimal agent $\pi^*(a_t|\tau_{0:t})$ which does not access to task labels w
  - but is assumed to have access to past observed interactions with the task, e.g. using a LSTM memory, solves: $\max_\pi \sum_w p(w) \sum_{\tau_0 0:\infty} p_\pi(\tau_{0:\infty}|w)[\sum_{t=0}^{\infty} \gamma^t r_t]$

- Assumptions of the POMDPs : Belief states
  - The belief state $b_t(x, w) = p(x, w|\tau_{0:t}) = \delta(x - x_t)p(w|\tau_{0:t})$, where $p(w|\tau_{0:t})$ is the posterior over tasks
  - The agent itself has no access to w, the posterior satisfies(See Appendix A),

Theorem

$$p(w|\tau_{0:t}) \propto p(w)p_0(x_0|w)\prod_{t'=0}^{t-1} P(x_{t'+1}|x_{t'}, a_{t'}, w)R(r_{t'}|x_{t'}, a_{t'}, x_{t'+1}), w).$$

$$p(w|\tau_{0:t}) \propto p(w)p_0(x_0|w)\prod_{t'=0}^{t-1} P(r_{t'}, x_{t'+1}|x_{t'}, a_{t'}, w)$$

  - That is, given $\tau_{0:t}$, the posterior is independent of the policy which generated the trajectory
  - This result will allow us to drive an off-policy algorithm
  - We overload notation and refer to the posterior alone as belief state $b_t(w) = p(w|\tau_{0:t})$, since it's the only interesting part

- So, how to train the model?
  - The optimal meta-learner only needs to make decisions based on current state x_t and the current belief $b_t(w)$
  - This implication being that we can restrict as; $\pi(a_t|\tau_{0:t}) \equiv \pi(a_t|x_t, b_t)$
  - But b_t is intractable to compute, requiring detailed knowledge of POMDPs conditional distributions
  - This argument motivates an agent consisting of two modules;

1. $\pi(a_t|x_t, \hat{b}_t)$: the policy dependent on the current state and (an approximate representation of)the posterior
2. the belief module : learns to output an approximate representation $\hat{b}_t$ of belief

Problem But how can we estimate(train) the $\hat{b}_t$?

Learning the Belief Network

- The problems of estimating belief posterior $\hat{b}_t$
  - Most past work has taken unsupervised approach to learning belief representations, because there are no access to additional useful information about the true underlying belief state
  - Learning belief modules via unsupervised approaches is difficult, and is general unsolved

- Auxiliary Supervised Learning
  - Fortunately, in meta-RL scenario, we can use the task information designed by human
  - Denote h_t as task information, w as task description(e.g. the location of the goal), $\tau_{0:t}$ as trajectory

Learning the Belief Network

- Different type of task description for Supervised Learning
  1. *Task description* : train a belief module $b_\theta(h_t|\tau_{0:t})$ to directly true task description $h_t = w$
  2. *Expert actions* : Assume we have expert agents $\pi^{w;e}(a_t|\tau_{0:t})$ on each training task, and we can train the belief module to predict the action chosen by expert, $h_t = a_t^{w;e}$
  3. *Task embeddings* : We can define K tasks and indexing these by $\{1, 2, ..., K\}$, so we can train belief module to predict the index i^w of task w, $h_t = i^w$
     - cf) if we have an embedding $F^w \in \mathbb{R}^d$ of each task using pre-training, we can predict $h_t = F^w$(See Appendix B)

  Note Once the belief module is learnt, this information is not needed during test time

Learning the Belief Network

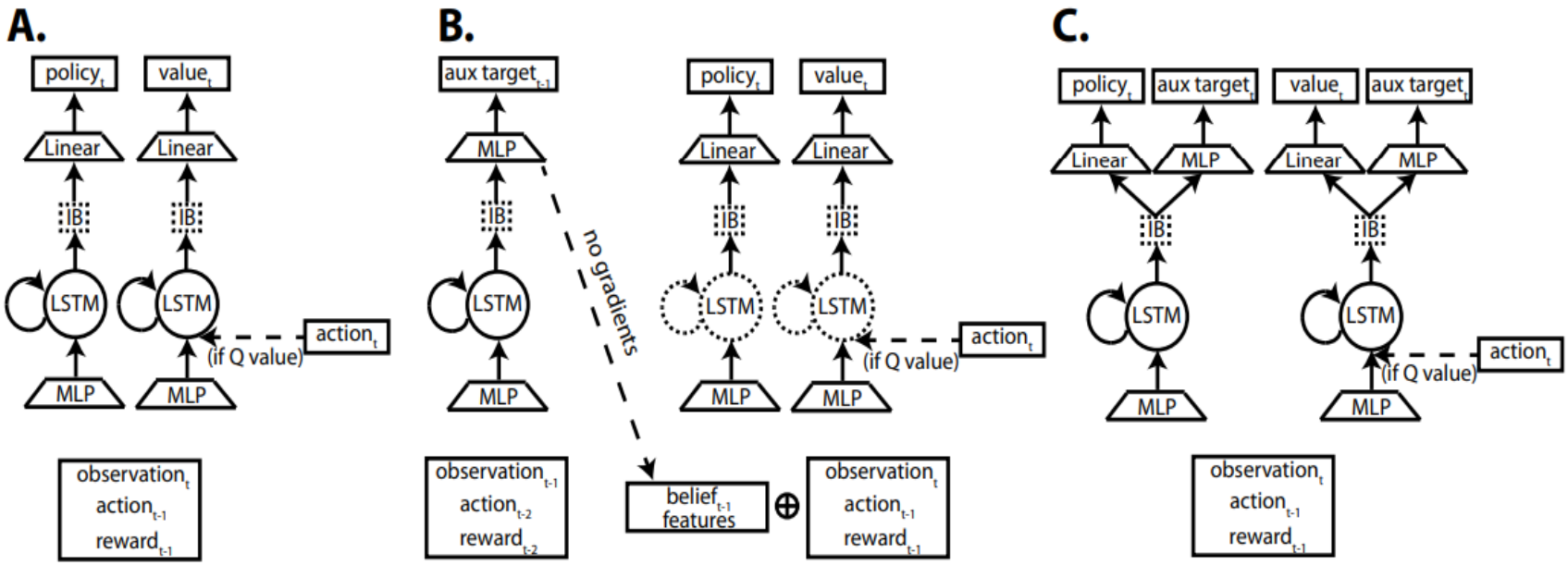
- Train a belief module as supervised learning
  - train belief modules $b_\theta(h_t|\tau_{0:t})$ to predict task information in a supervised way by minimizing $\mathbb{E}_{p(h_t|\tau_{0:t})}[-\log b_\theta(h_t|\tau_{0:t})]$
  - So, the target distribution is $p(h_t|\tau_{0:t})$, we can minimize $\mathbb{KL}(p(h_t|\tau_{0:t})||b_\theta(h_t|\tau_{0:t}))$

Note

In this meta-RL setup(c.f. amortized inference, Samuel, Gershman et al.,2014), we can obtain samples from posterior because each training episode, because $(w, \tau_{0:t}) \sim p(w)p_\pi(\tau_{0:t}|w)$

The posterior over w and h_t is independent of the policy, belief network can be trained using off-policy data generated by previous policies

Architectures and Algorithms



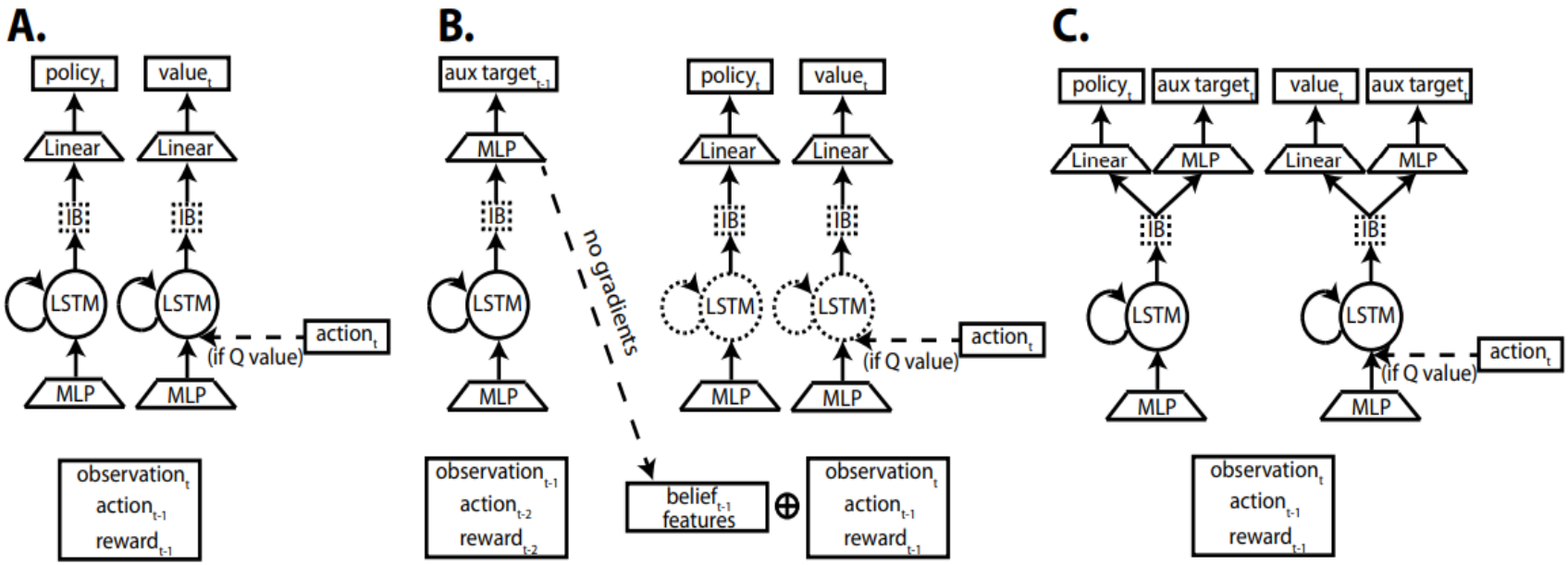
A : A baseline LSTM agent

B : Belief network agent

C : Auxiliary head agent

* LSTMs and information bottleneck(IB) are optional
* RL Algorithms : SVG(0)(Nicolas Heess et al.,2015) as off-policy, PPO(John Schulman, et al.,2017) as on-policy

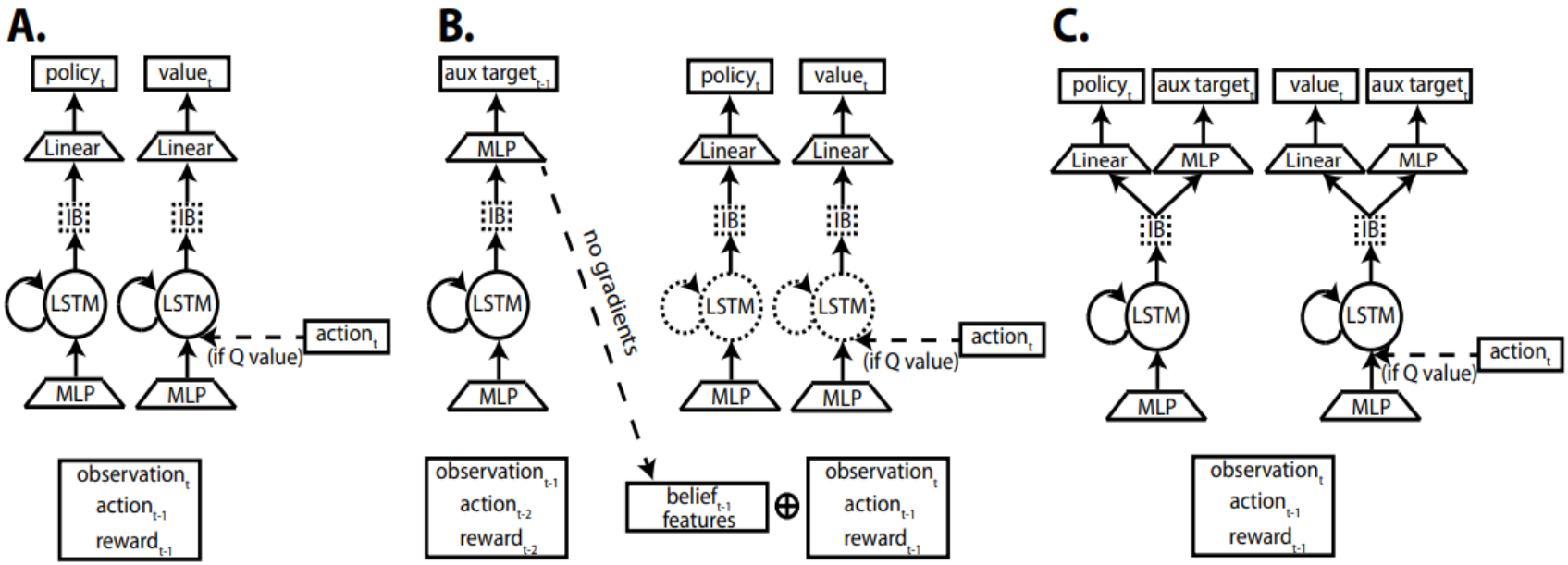
Architectures and Algorithms



A : A baseline LSTM agent

- A general recent meta-RL formulation without belief networks
- Similar to RL^2(Yan Duan, et al.,2017)

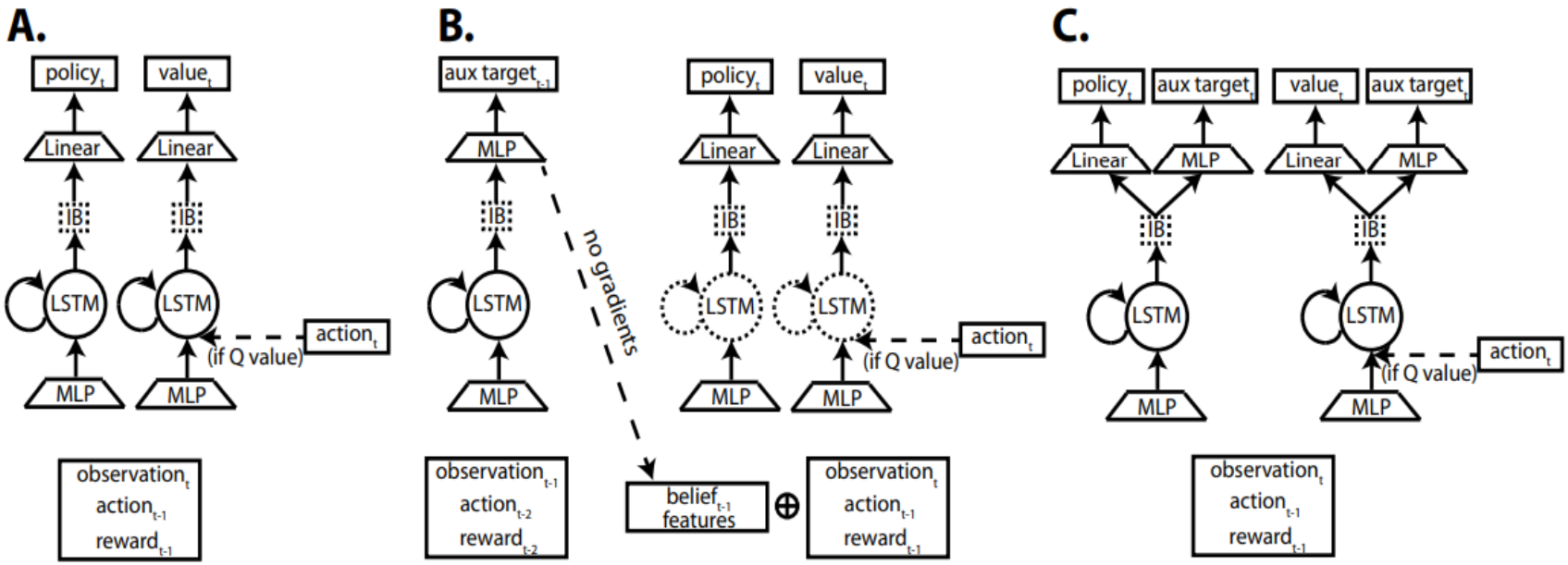
Architectures and Algorithms



B : Belief network agent

- Augment the baseline with a belief net which output an approximate posterior over the auxiliary task information h_t
- The actor and critic are fed a representation of the belief state given by the penultimate layer(performance issue)
- Do not backpropagate gradients to the belief net $\rightarrow$ enable that belief net focusing on learn good representation itself

Architectures and Algorithms



C : Auxiliary head agent

- The auxiliary supervised belief loss directly shapes the representations inside the actor and critic networks

* Information Bottleneck(IB)

- Regularization the learnt representation by adding a stochastic layer(Deep variational information bottleneck, Alexander A Alemi, et al.,2017) on top of the LSTMs

Experiments

- Experiments focus on;
  1. Direct training of belief net with previleged information can speed up learning
  2. Train the recurrent agents efficiently off-policy
  3. IB regularization is an effective way for speeding up off-policy learning(PPO agents do not use IB)
  4. Scale up to complex continuous env(Numpad) with sparse rewards and requiring long-term memory

- environments
  - Multi-armed bandit, Semicircle, Cheetah velocity, Noisy target, Numpad(See Appendix)
  - Train on 100 tasks and evaluate on a holdout set of 1000 tasks

Experiments

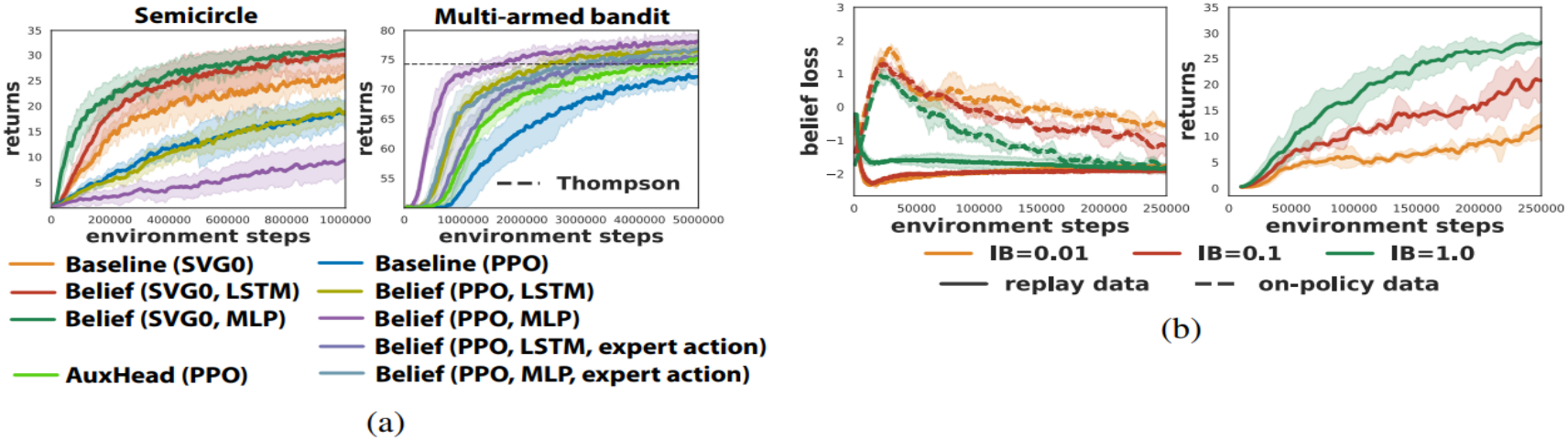


Figure 2: **(a)** Comparison of off-policy SVG(0) and on-policy PPO algorithms, and a demonstration of the advantage of supervision. **(b)** Increased IB regularization strength in the belief network leads to smaller generalization gap (left) and better performance (right) in the search for target task.

- Multi-armed bandit
  - 20 arms and 100 horizon
  - *task description* : a vector of arm probabilities
- Semicircle
  - A point mass has to find a target on a semicircle
  - *task description* : an angle of semi-circle(e.g. max $= 2\pi$)

Experiments

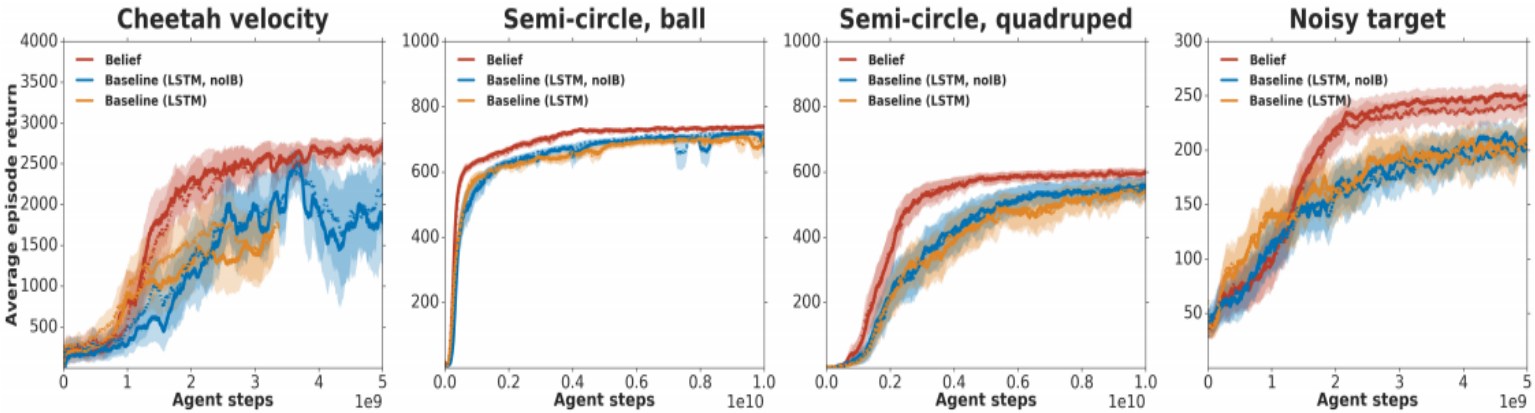
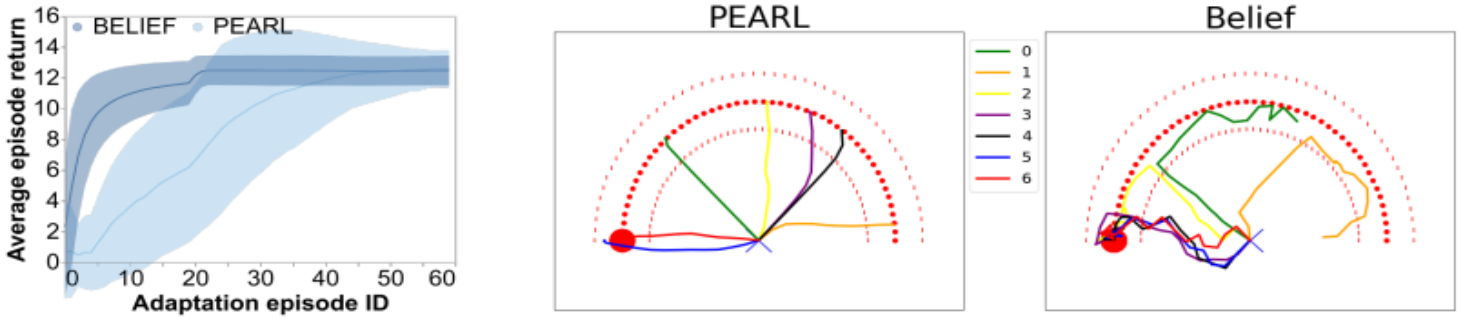


Figure 3: A summary of the advantage of using the belief network supervised with task descriptions. Solid curves correspond to performance on training tasks, while dashed curves to performance on validation tasks.

- Cheetah velocity
  - $r(v) = \max(1 - |\frac{v}{v_{\text{target}}} - 1|, 0)$, *task description* : v_{target}
- Noisy target
  - rolling ball but not teleported to origin upon reaching target
  - Should focus on integrate Bernoulli noisy reward(being 1 $p = \min(0.5, (1 + d^2)^{-1})$) rather then remembering target

Experiments

- Behavioral analysis



(a) (b)

Figure 6: **(a)** Comparison of the average episodic returns during adaptation of our Belief agent and PEARL agent in the sparse 2d navigation task from [14]. The error bars show the variance of the adaptation curves across tasks and seeds (1 standard deviation). **(b)** An example adaptation behavior of Belief and PEARL agents in the sparse 2d navigation domain. The red dot represents the target the agent searches for. Different colors correspond to different episodes during adaptation.

* PEARL agent vs Belief agent in 2d navigation
  - The belief agent explores as much of the semicircle as possible in every episode until it finds the rewarding target
  - The PEARL agent simply tries a single target on the semicircle in every episode, and repeats this sampling until it hits the actual rewarding target

Conclusion

- Main contributions
  1. We demonstrate that leveraging this cheap task information during meta-training is a simple and cheap way to boost the performance of meta-RL agents
  2. We show that we can train meta-RL agents with recurrent policies efficiently using off-policy algorithms
  3. We experimentally demonstrate that our agents can solve difficult meta-RL problems in continuous control environments, involving sparse rewards

References

- Meta-Learning(Computer Science), Wikipedia
- Partially observable Markov decision process, Wikipedia
- Meta-Learning: from Few-Shot Learning to Rapid Reinforcement Learning, ICML 2019 Tutorial
- Ravi, Sachin, and Hugo Larochelle. "Optimization as a model for few-shot learning." (2016).
- Rakelly, Kate, et al. "Efficient off-policy meta-reinforcement learning via probabilistic context variables." arXiv preprint arXiv:1903.08254 (2019).

Thank you for your attention!