DRL Seminar: Meta-Learning

Steven Battilana
Supervisor: Giambattista Parascandolo

ETH zürich

DRL Seminar: Meta-Learning (part 2)

Steven Battilana

Supervisor: Giambattista Parascandolo

Outline

Meta-Learning in RL[18]

Reinforcement learning:

$$\theta^{\star} = arg\max_{\theta} E_{\pi_{\theta}(\tau)}[R(\tau)]$$
$$= f_{\mathsf{RL}}(\mathcal{M}),$$

where $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, r\}$ is the MDP.

Meta-reinforcement learning:

$$\theta^{\star} = arg\max_{\theta} \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)],$$

where $\phi_i = f_{\theta}(\mathcal{M}_i)$ is the MDP for task i.

[18] Duan et al. (2016)

Meta-Gradient RL

Meta-Gradient RL

- Online meta-learning of the discount factor γ

Meta-Gradient RL

- Online meta-learning of the discount factor γ
- Achieved new state-of-the-art performance (at the time of publication)

Meta-Gradient RL

- Online meta-learning of the discount factor γ
- Achieved new state-of-the-art performance (at the time of publication)
- Take away: Meta-learning the discount factor γ instead of parameter θ

Meta-Learning in RL[18]

Reinforcement learning:

$$\theta^\star = arg \max_\theta E_{\pi_\theta(\tau)}[R(\tau)]$$
$$= f_{\text{RL}}(\mathcal{M}),$$

where $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, r\}$ is the MDP.

Meta-reinforcement learning:

$$\theta^\star = arg \max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)],$$

where $\phi_i = f_\theta(\mathcal{M}_i)$ is the MDP for task i.

[18] Duan et al. (2016)

Meta-Learning in RL[18]

Reinforcement learning:

$$\theta^\star = arg \max_\theta E_{\pi_\theta(\tau)}[R(\tau)]$$

$$= arg \max_\theta E_{\pi_\theta(\tau)}\left[\sum_{t=0}^{T} \gamma^t r(s_t, \pi_\theta(a_t|s_t))\right]$$

where $\gamma \in [0,1]$ is the discount factor, r is the reward function.

Meta-reinforcement learning:

$$\theta^\star = arg \max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)],$$

where $\phi_i = f_\theta(\mathcal{M}_i)$ is the MDP for task i.

[18] Duan et al. (2016)

Meta-Gradient RL: General Algorithm Overview[8]

(i) Update neural network parameters θ

[17] Sutton (1992)
[8] Xu et al. (2018)

Meta-Gradient RL: General Algorithm Overview[8]

(i) Update neural network parameters θ

(ii) Online cross-validation[17] using the meta-objective J'

[17] Sutton (1992)
[8] Xu et al. (2018)

Meta-Gradient RL: General Algorithm Overview[8]

(i) Update neural network parameters θ

(ii) Online cross-validation[17] using the meta-objective J'

(iii) Update meta-parameters η

[17] Sutton (1992)
[8] Xu et al. (2018)

Meta-Gradient RL: General Algorithm Overview[8]

(i) Update neural network parameters θ

(ii) Online cross-validation[17] using the meta-objective J'

(iii) Update meta-parameters η

[17] Sutton (1992)

[8] Xu et al. (2018)

Meta-Gradient RL: General Algorithm Detail[8]

(i) • ***Update function***

$$\theta' = \theta + f(\tau, \theta, \eta),$$

[101] cf. Appendix D
[8] Xu et al. (2018)

Meta-Gradient RL: General Algorithm Detail[8]

(i) • ***Update function***

$$\theta' = \theta + f(\tau, \theta, \eta),$$

where $\tau_t = \{s_t, a_t, r_{t+1}, ...\}$ consisting of states s, actions a, and rewards r; current neural network parameters θ; ***meta-parameters*** $\eta_t = \{\gamma_t, \lambda_t\}$.

[101] cf. Appendix D
[8] Xu et al. (2018)

Meta-Gradient RL: General Algorithm Detail[8]

(i)

- ***Update function***

$$\theta' = \theta + f(\tau, \theta, \eta),$$

  where $\tau_t = \{s_t, a_t, r_{t+1}, ...\}$ consisting of states s, actions a, and rewards r; current neural network parameters θ; ***meta-parameters*** $\eta_t = \{\gamma_t, \lambda_t\}$.
- $f(\tau, \theta, \eta)$ could be an update function that applies SGD to update the agent's parameters θ. E.g. *A2C objective* semi-gradient or *squared error* semi-gradient[101].

[101] cf. Appendix D

[8] Xu et al. (2018)

Meta-Gradient RL: General Algorithm Overview[8]

(i) Update neural network parameters θ

(ii) Online cross-validation[17] using the meta-objective J'

(iii) Update meta-parameters η

---

[17] Sutton (1992)

[8] Xu et al. (2018)

Meta-Gradient RL: General Algorithm Overview[8]

(i) Update neural network parameters θ

(ii) Online cross-validation[17] using the meta-objective J'

(iii) Update meta-parameters η

[17] Sutton (1992)
[8] Xu et al. (2018)

Meta-Gradient RL: General Algorithm Detail[8]

(ii) ***Online cross-validation***[17], i.e. compute agent's performance using the ***meta-objective***

$$J'(\tau', \theta', \eta'),$$

[17] Sutton (1992)

[102] $J'(\tau', \theta', \eta')$ examples for prediction and control are in Appendix D

[8] Xu et al. (2018)

Meta-Gradient RL: General Algorithm Detail[8]

(ii) ***Online cross-validation***[17], i.e. compute agent's performance using the ***meta-objective***

$$J'(\tau', \theta', \eta'),$$

where $\tau'_T = \{s'_T, a'_T, r'_{T+1}, ...\}$ consisting of states s, actions a, and rewards r; updated neural network parameters θ' from (i); meta-parameters $\eta' = \{\gamma', \lambda'\}, \gamma' = \lambda' = 1$ (long-sighted return)[102].

[17] Sutton (1992)
[102] $J'(\tau', \theta', \eta')$ examples for prediction and control are in Appendix D
[8] Xu et al. (2018)

Meta-Gradient RL: General Algorithm Overview[8]

(i) Update neural network parameters θ

(ii) Online cross-validation[17] using the meta-objective J'

(iii) Update meta-parameters η

[17] Sutton (1992)

[8] Xu et al. (2018)

Meta-Gradient RL: General Algorithm Overview[8]

(i) Update neural network parameters θ

(ii) Online cross-validation[17] using the meta-objective J'

(iii) Update meta-parameters η

[17] Sutton (1992)

[8] Xu et al. (2018)

Meta-Gradient RL: General Algorithm Detail[8]

(iii) ***Update meta-parameters*** η

$$\eta_T = \eta_t - \beta \frac{\partial}{\partial \theta'} J'(\tau', \theta', \eta') z'(\theta, f(\tau, \theta, \eta))$$

[103] η_T examples for prediction and control are in Appendix D
[8] Xu et al. (2018)

Meta-Gradient RL: General Algorithm Detail[8]

(iii) ***Update meta-parameters*** η

$$\eta_T = \eta_t - \beta \frac{\partial}{\partial \theta'} J'(\tau', \theta', \eta') z'(\theta, f(\tau, \theta, \eta))$$

where $\tau_t = \{s_t, a_t, r_{t+1}, ...\}$, $\tau'_T = \{s'_T, a'_T, r'_{T+1}, ...\}$ consisting of states s, actions a, and rewards r; current neural network parameters θ; updated neural network parameters θ' from (i); meta-parameters $\eta_t = \{\gamma_t, \lambda_t\}$, $\eta' = \{\gamma', \lambda'\}, \gamma' = \lambda' = 1$.[103]

[103] η_T examples for prediction and control are in Appendix D

[8] Xu et al. (2018)

Meta-Gradient RL Results[8]

- Comparing against Rainbow the state-of-the-art agent trained on Atari games[12].

	Human starts	No-op starts
Rainbow	153%	223%
Meta-Gradient	293%	288%

Table: Median human-normalised score using 200M frames.

[12] Hessel (2017)
[8] Xu et al. (2018)

Meta-Learning in RL[18]

Reinforcement learning:

$$\theta^\star = arg\max_\theta E_{\pi_\theta(\tau)}\left[\sum_{t=0}^{T} \gamma^t r(s_t, \pi_\theta(a_t|s_t))\right]$$

where $\gamma \in [0, 1]$ is the discount factor, r is the reward function.

Meta-reinforcement learning:

$$\theta^\star = arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)],$$

where $\phi_i = f_\theta(\mathcal{M}_i)$ is the MDP for task i.

[18]Duan et al. (2016)

Meta-Learning in RL[18]

Reinforcement learning:

$$\theta^\star = arg\max_\theta E_{\pi_\theta(\tau)}[R(\tau)]$$

where $\gamma \in [0,1]$ is the discount factor, r is the reward function.

Meta-reinforcement learning:

$$\theta^\star = arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}\left[\sum_{t=0}^{T} \gamma^t r(s_t, \pi_\theta(a_t|s_t))\right],$$

where $\phi_i = f_\theta(\mathcal{M}_i)$ is the MDP for task i.

[18] Duan et al. (2016)

Meta-Learning Optimisation Problem (Reminder)[15]

(A) Meta-Training:

(i) Meta-Learning:

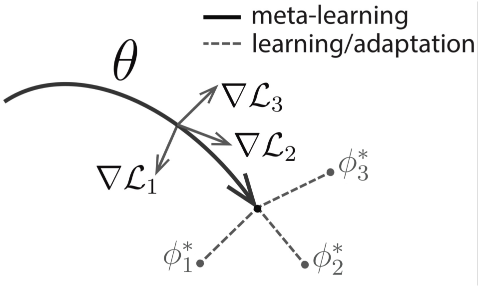
$$\theta^\star = arg\max_{\theta} \log(p(\theta|\mathcal{D}_{\text{meta-train}}))$$

(ii) Adaption (per training task):

$$\phi^\star = arg\max_{\phi} \log(p(\phi|\mathcal{D}^{\text{train}}, \theta^\star))$$

(B) Meta-Testing:

$$\phi^* = arg\max_{\phi} \log(p(\phi|\mathcal{D}_{\text{meta-test}}, \theta^\star))$$



[15] Finn (2019)

MAML Criticism: Mutually-Exclusive Task Distributions[4]

Figure: An example of *mutually-exclusive* task distributions. In each task of mutually-exclusive few-shot classification, different classes are randomly assigned to the N-way classification labels. The same class, such as the dog and butterfly in this illustration, can be assigned different labels across tasks which makes it impossible for one model to solve all tasks simultaneously.

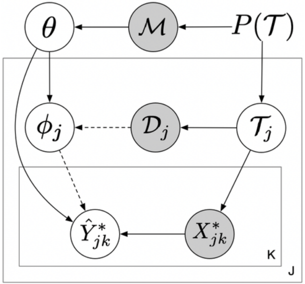


Figure: Graphical model for meta-learning. Observed variables are shaded. Without either one of the dashed arrows, $\hat{Y}^*$ is conditionally independent of $\mathcal{D}$ given θ and X^*, which we refer to as complete memorisation.

[4]Yin et al. (2019)

Meta-Regularised Model-Agnostic Meta-Learning (MR-MAML)

Meta-Regularised Model-Agnostic Meta-Learning (MR-MAML)

- Introducing (meta-)regularisation to MAML

Meta-Regularised Model-Agnostic Meta-Learning (MR-MAML)

- Introducing (meta-)regularisation to MAML
- Extends to non-mutually exclusive tasks

Meta-Regularised Model-Agnostic Meta-Learning (MR-MAML)

- Introducing (meta-)regularisation to MAML
- Extends to non-mutually exclusive tasks
- Take away: enables MAML to learn on non-mutually exclusive tasks

Meta-Learning in RL[18]

Reinforcement learning:

$$\theta^\star = arg\max_\theta E_{\pi_\theta(\tau)}[R(\tau)]$$

where $\gamma \in [0, 1]$ is the discount factor, r is the reward function.

Meta-reinforcement learning:

$$\theta^\star = arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)} \left[\sum_{t=0}^{T} \gamma^t r(s_t, \pi_\theta(a_t|s_t)) \right],$$

where $\phi_i = f_\theta(\mathcal{M}_i)$ is the MDP for task i.

[18] Duan et al. (2016)

Meta-Regularised MAML[4]

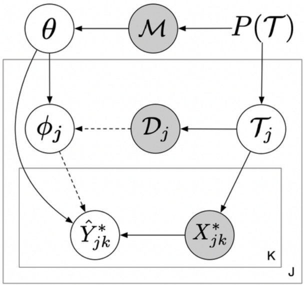


Figure: Graphical model for meta-learning. Observed variables are shaded. Without either one of the dashed arrows, $\hat{Y}^*$ is conditionally independent of $\mathcal{D}$ given θ and X^*, which we refer to as complete memorisation.

- Meta-RL: $\theta^\star = arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)]$

[4] Yin et al. (2019)

Meta-Regularised MAML[4]

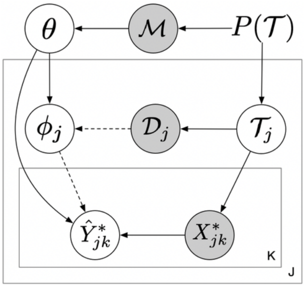


Figure: Graphical model for meta-learning. Observed variables are shaded. Without either one of the dashed arrows, $\hat{Y}^*$ is conditionally independent of $\mathcal{D}$ given θ and X^*, which we refer to as complete memorisation.

- Meta-RL: $\theta^\star = arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)]$
- Approximated policy π_{ϕ_i}, $\phi_i = f_\theta(\mathcal{M}_i)$

[4]Yin et al. (2019)

Meta-Regularised MAML[4]

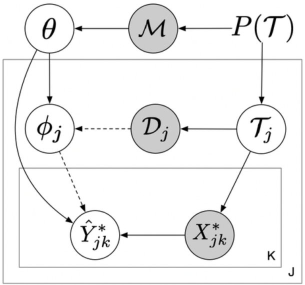


Figure: Graphical model for meta-learning. Observed variables are shaded. Without either one of the dashed arrows, $\hat{Y}^*$ is conditionally independent of $\mathcal{D}$ given θ and X^*, which we refer to as complete memorisation.

- Meta-RL: $\theta^\star = arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)]$
- Approximated policy π_{ϕ_i}, $\phi_i = f_\theta(\mathcal{M}_i)$
- MAML meta-update: $\theta' = \theta + \nabla_\theta \mathcal{L}_\mathcal{T}(f_\theta)$

[4] Yin et al. (2019)

Meta-Regularised MAML[4]

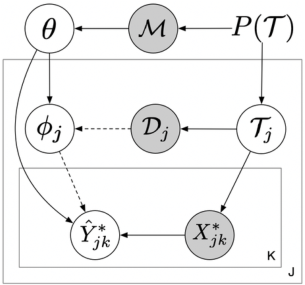


Figure: Graphical model for meta-learning. Observed variables are shaded. Without either one of the dashed arrows, $\hat{Y}^*$ is conditionally independent of $\mathcal{D}$ given θ and X^*, which we refer to as complete memorisation.

- Meta-RL: $\theta^\star = arg\max_\theta \sum_{i=1}^n E_{\pi_{\phi_i}(\tau)}[R(\tau)]$
- Approximated policy π_{ϕ_i}, $\phi_i = f_\theta(\mathcal{M}_i)$
- MAML meta-update: $\theta' = \theta + \nabla_\theta \mathcal{L}_\mathcal{T}(f_\theta)$
- **MR-MAML meta-update:**

$$\theta' = \theta + \nabla_\theta(\mathcal{L}_\mathcal{T}(f_\theta) + D_{\mathrm{KL}}(q(\theta|\mathcal{M})||r(\theta))),$$

[4] Yin et al. (2019)

Meta-Regularised MAML[4]

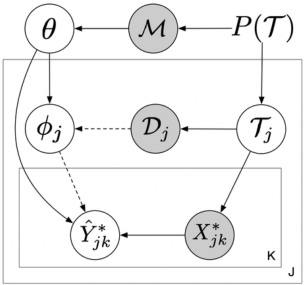


Figure: Graphical model for meta-learning. Observed variables are shaded. Without either one of the dashed arrows, $\hat{Y}^*$ is conditionally independent of $\mathcal{D}$ given θ and X^*, which we refer to as complete memorisation.

- Meta-RL: $\theta^\star = arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)]$
- Approximated policy π_{ϕ_i}, $\phi_i = f_\theta(\mathcal{M}_i)$
- MAML meta-update: $\theta' = \theta + \nabla_\theta \mathcal{L}_\mathcal{T}(f_\theta)$
- **MR-MAML meta-update:**

$$\theta' = \theta + \nabla_\theta(\mathcal{L}_\mathcal{T}(f_\theta) + D_{\mathrm{KL}}(q(\theta|\mathcal{M})||r(\theta))),$$

where $q(\theta|\mathcal{M})$ summarises meta-training data into a distribution on meta-parameters, and $r(\theta)$ is a variational approximation to the marginal, which is set to $\mathcal{N}(\theta; 0, I)$.

[4] Yin et al. (2019)

Non-Mutually-Exclusive (NME) Datasets[4]

In the non-mutually-exclusive N-way K-shot classification problem, each class is randomly assigned a fixed classification label from 1 to N.

Figure: An example of *mutually-exclusive* task distributions. In each task of non-mutually-exclusive few-shot classification, different classes are randomly assigned to the fixed N-way classification labels. The same class, such as the dog and butterfly in this illustration, are assigned to the same labels across tasks which makes it possible for one model to solve all tasks simultaneously.

[4]Yin et al. (2019)

Experiments[4]

Meta-test *pre-update* accuracy:

NME Omniglot	20-way 1-shot	20-way 5-shot
MAML	99.2 (0.2)%	45.1 (38.9)%
TAML	68.9(43.1)%	6.7 (1.8)%
MR-MAML (ours)	**5.0 (0)**%	**5.0 (0)**%

Figure: Table 5: **Meta-test *pre-update* accuracy** on non-mutually-exclusive (NME) classification. MR-MAML controls the meta-training pre-update accuracy close to random guess and achieves low training error after adaptation.

Meta-test accuracy:

NME Omniglot	20-way 1-shot	20-way 5-shot
MAML	7.8 (0.2)%	50.7 (22.9)%
TAML (Jamal & Qi, 2019)	9.6 (2.3)%	67.9 (2.3)%
MR-MAML (W) (ours)	**83.3 (0.8)**%	**94.1 (0.1)**%

Figure: Table 4: **Meta-test accuracy** on non-mutually-exclusive (NME) classification. The fine-tuning and nearest neighbour baseline results for mini-ImageNet are from [7] Ravi and Larochelle (2016).

[4] Yin et al. (2019)

Datasets for Meta-Learning

- Dataset MAML (cheetah and ant locomotion tasks), ANIL (Omniglot, MiniImagenet), MR-MAML (Omniglot, MiniImagenet)

Datasets for Meta-Learning

- Dataset MAML (cheetah and ant locomotion tasks), ANIL (Omniglot, MiniImagenet), MR-MAML (Omniglot, MiniImagenet)
- Dataset RL2 (ViZDoom environment)

Datasets for Meta-Learning

- Dataset MAML (cheetah and ant locomotion tasks), ANIL (Omniglot, MiniImagenet), MR-MAML (Omniglot, MiniImagenet)
- Dataset RL2 (ViZDoom environment)
- Meta-Gradient RL used for instance on Atari video games

Meta-World

Meta-World

- An open-source simulated benchmark of 50 distinct robotic manipulations tasks

Meta-World

- An open-source simulated benchmark of 50 distinct robotic manipulations tasks
- Introducing a benchmark for meta-learning to challenge researchers

Meta-World

- An open-source simulated benchmark of 50 distinct robotic manipulations tasks
- Introducing a benchmark for meta-learning to challenge researchers
- Take away: Benchmark for meta-learning for future approaches

Meta-World[5]

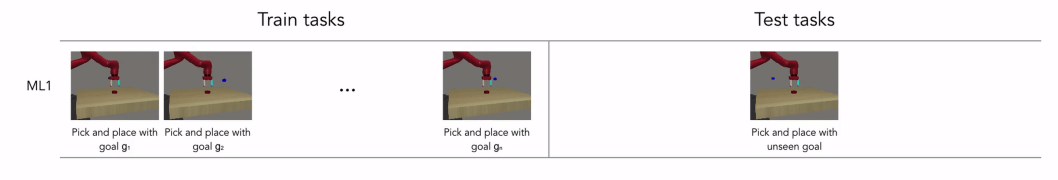


Figure: Visualisation of within task adaptation in ML1.

[5]Yu et al. (2019)

Meta-World[5]

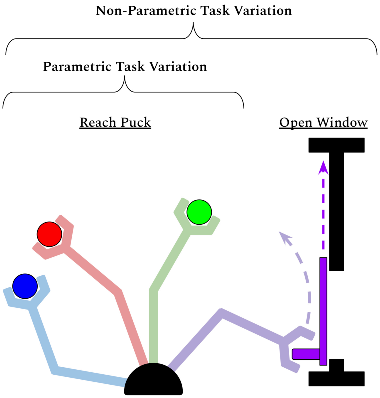


Figure: Parametric/non-parametric variation: all "reach puck" tasks (left) can be parametrised by the puck position, while the difference between "reach puck" and "open window" (right) is non-parametric.

[5] Yu et al. (2019)

Meta-World[5]

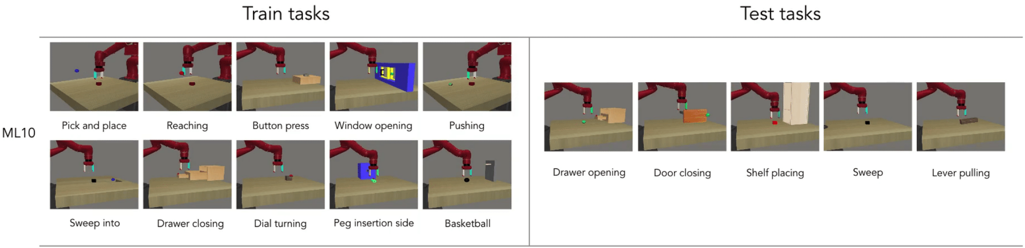


Figure: Visualisation of adapting to new tasks in ML10.

[5] Yu et al. (2019)

Meta-World[5]

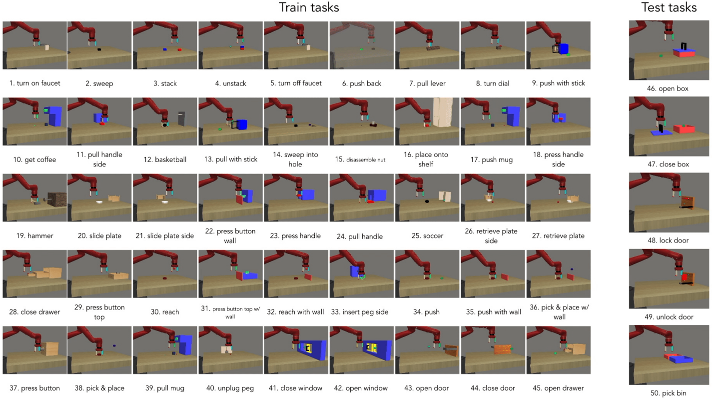


Figure: Visualisation of adapting to new tasks in ML45.

[5] Yu et al. (2019)

Meta-World[5]

- Task distributions are very narrow.

[5] Yu et al. (2019)

Meta-World[5]

- Task distributions are very narrow.
- Propose open-source benchmark for meta-RL.

[5] Yu et al. (2019)

Meta-World[5]

- Task distributions are very narrow.
- Propose open-source benchmark for meta-RL.
- Large number of simulated robotic manipulation tasks.

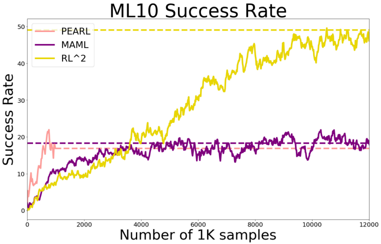
[5] Yu et al. (2019)

Meta-World[5]

- Task distributions are very narrow.
- Propose open-source benchmark for meta-RL.
- Large number of simulated robotic manipulation tasks.
- Introducing three different difficulty modes for evaluation (ML1, ML10, ML45).

[5] Yu et al. (2019)

Meta-World[5]



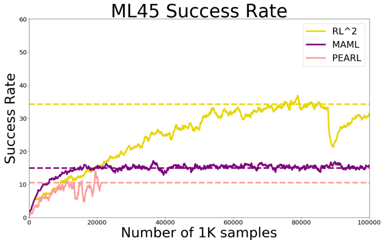


Figure: Learning curves of all methods on ML10, and ML45 benchmarks. Y-axis represents success rate averaged over tasks in percentage (%). The dashed lines represent asymptotic performances.

Methods	ML10		ML45	
	meta-train	meta-test	meta-train	meta-test
MAML	25%	**36%**	21.14%	23.93%
RL^2	**50%**	10%	**43.18%**	20%
PEARL	42.78%	0%	11.36%	**30%**

Figure: Average success rates over all tasks for ML10, and ML45. The best performance in each benchmark is bolden. For ML10 and ML45, we show the meta-train and meta-test success rates.

[5] Yu et al. (2019)

Conclusion

Conclusion

- Meta-Learning Flavours (part 1 and 2)

Conclusion

- Meta-Learning Flavours (part 1 and 2)
  (i) Meta-learn ideal initialisation parameters θ for policy $\pi_\theta(a|s)$ (MAML, ANIL, MR-MAML)

Conclusion

- Meta-Learning Flavours (part 1 and 2)
  (i) Meta-learn ideal initialisation parameters θ for policy $\pi_\theta(a|s)$ (MAML, ANIL, MR-MAML)
  (ii) Meta-learn ideal optimiser (Learning to learn by gradient descent by gradient descent)

Conclusion

- Meta-Learning Flavours (part 1 and 2)
  - (i) Meta-learn ideal initialisation parameters θ for policy $\pi_\theta(a|s)$ (MAML, ANIL, MR-MAML)
  - (ii) Meta-learn ideal optimiser (Learning to learn by gradient descent by gradient descent)
  - (iii) Meta-learn policy $\pi_\theta(a|s)$ using a RNN (RL^2)

Conclusion

- Meta-Learning Flavours (part 1 and 2)
  (i) Meta-learn ideal initialisation parameters θ for policy $\pi_\theta(a|s)$ (MAML, ANIL, MR-MAML)
  (ii) Meta-learn ideal optimiser (Learning to learn by gradient descent by gradient descent)
  (iii) Meta-learn policy $\pi_\theta(a|s)$ using a RNN (RL^2)
  (iv) Meta-learn (online) ideal discount factor γ (Meta-Gradient RL)

Conclusion

- Meta-Learning Flavours (part 1 and 2)
  (i) Meta-learn ideal initialisation parameters θ for policy $\pi_\theta(a|s)$ (MAML, ANIL, MR-MAML)
  (ii) Meta-learn ideal optimiser (Learning to learn by gradient descent by gradient descent)
  (iii) Meta-learn policy $\pi_\theta(a|s)$ using a RNN (RL^2)
  (iv) Meta-learn (online) ideal discount factor γ (Meta-Gradient RL)
- Meta-World benchmark challenging Meta RL researchers

Reference I

[1] Hae Beom Lee, Hayeon Lee, Donghyun Na, Saehoon Kim, Minseop Park, Eunho Yang, and Sung Ju Hwang. *Learning to Balance: Bayesian Meta-Learning for Imbalanced and Out-of-distribution Tasks*. 2019.
`arxiv.org/abs/1905.12917`

[2] Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. *Human-level concept learning through probabilistic program induction*. 2015.
`mit.edu/Science-2015-Lake-1332-8.pdf`

[3] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. *Meta-learning for semi-supervised few-shot classification*. 2018.
`arxiv.org/abs/1803.00676`

[4] Mingzhang Yin, George Tucker, Mingyuan Zhou, Sergey Levine, and Chelsea Finn. *Meta-Learning without Memorization*. 2019.
`arxiv.org/abs/1912.03820`

Reference II

[5] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. *Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning*. 2019.
`arxiv.org/abs/1910.10897`

[6] Chelsea Finn, Pieter Abbeel, and Sergey Levine. *Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks*. 2017.
`arxiv.org/abs/1703.03400`

[7] Sachin Ravi and Hugo Larochelle. *Optimization as a model for few-shot learning*. 2016.
`openreview.net/RaviLarochelle`

[8] Zhongwen Xu, Hado van Hasselt, and David Silver. *Meta-Gradient Reinforcement Learning*. 2018.
`arxiv.org/abs/1805.098017`

[9] Aldo Faisal. *Imperial college London 424h – learning in autonomous systems (reinforcement learning)*. 2018.

[10] Sergios Karagiannakos. *The idea behind Actor-Critics and how A2C and A3C improve them*. 2018.
`theaisummer.com/Actor_critics`
Accessed: 04/04/2020

Reference III

[11] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. *Universal value function approximators*. 2015.
proceedings.mlr.press/v37/schaul15

[12] Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Daniel Horgan, Bilal Piot, Mohammad Gheshlaghi Azar, and David Silver. *Rainbow: Combining Improvements in Deep Reinforcement Learning*. 2017.
arxiv.org/abs/1710.02298

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention Is All You Need*. 2017.
arxiv.org/abs/1706.03762

[14] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. *ImageNet Large Scale Visual Recognition Challenge*. 2014.
arxiv.org/abs/1409.0575

Reference IV

[15] Chelsea Finn. *Stanford CS 330: Deep Multi-Task and Meta Learning (Fall 2019)*. 2019.
`cs330.stanford.edu`
Accessed: 22/04/2020

[16] Adrien Lucas Ecoffet. *An Intuitive Explanation of Policy Gradient*. 2018.
`towardsdatascience.com/policy-gradient-intuition`
Accessed: 19/04/2020

[17] Richard S. Sutton. *Adapting bias by gradient descent: An incremental version of delta-bar-delta*. 1992.
`aaai.org/Papers/AAAI92-027.pdf`

[18] Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. *RL^2: Fast Reinforcement Learning via Slow Reinforcement Learning*. 2016.
`arxiv.org/abs/1611.02779`

[19] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. *Trust Region Policy Optimisation*. 2017.
`arxiv.org/abs/1502.05477`

Reference V

[20] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W. Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando de Freitas. *Learning to learn by gradient descent by gradient descent*. 2016. arxiv.org/abs/1606.04474

Appendix

B. Meta-Regularised MAML: The Memorisation Problem in Meta-Learning[4] I

Definition (Mutual Information, source: PAI)

$$I(X_i, X_j) := \sum_{x_i, x_j} P(x_i, x_j) \log \left(\frac{P(x_i, x_j)}{P(x_i)P(x_j)} \right).$$

Definition (Complete Meta-Learning Memorisation)

Complete memorisation in meta-learning is when the learned model ignores the task training data such that $I(\hat{y}^*; \mathcal{D}|x^*, \theta)$ = 0 (i.e., $q(\hat{y}^*|x^*, \theta) = E_{\mathcal{D}'|x^*}[q(\hat{y}^*|x^*, \theta, \mathcal{D}')]$).
Note: $\hat{y}^*, x^*$ is a test sample from the meta-training set.

[4] Yin et al. (2019)

B. Meta-Regularised MAML: The Memorisation Problem in Meta-Learning[4] II

Definition (KL-Divergence)

$$D_{\mathsf{KL}}(q(\theta|\mathcal{M})||r(\theta)) = \int q(\theta|\mathcal{M}) \log\left(\frac{q(\theta)|\mathcal{M})}{r(\theta)}\right) d\theta.$$

Lemma (Upper bound)

$$I(y^*_{1:N}, \mathcal{D}_{1:N}; \theta|x^*_{1:N}) = E\left[\log\left(\frac{q(\theta|\mathcal{M})}{q(\theta|x^*_{1:N})}\right)\right] \leq E[D_{\mathit{KL}}(q(\theta|\mathcal{M})||r(\theta)].$$

[4] Yin et al. (2019)

B. Meta-Regularised MAML: Algorithm[4,6] I

Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters

```
randomly initialize θ
while not done do
    Sample batch of tasks T_i ~ p(T)
    for all T_i do
        Evaluate ∇_θ L_{T_i}(f_θ) with respect to K examples
        Compute adapted parameters with gradient descent: θ'_i = θ − α∇_θ L_{T_i}(f_θ)
    end for
    Update θ ← θ − β∇_θ Σ_{T_i~p(T)} L_{T_i}(f_{θ'_i})
end while
```

Algorithm 2: Meta-Regularized MAML

input : Task distribution $p(\mathcal{T})$; Weights distribution $q(\theta;\tau) = \mathcal{N}(\theta;\tau)$ with Gaussian parameters $\tau = (\theta_\mu, \theta_\sigma)$; Prior distribution $r(\theta)$ and Lagrangian multiplier β; Stepsize α, α'.
output: Network parameter τ, $\tilde{\theta}$.

Initialize τ, $\tilde{\theta}$ randomly;
while *not converged* **do**
- Sample a mini-batch of $\{\mathcal{T}_i\}$ from $p(\mathcal{T})$;
- Sample $\theta \sim q(\theta;\tau)$ with reparameterization ;
- **for** *all* $\mathcal{T}_i \in \{\mathcal{T}_i\}$ **do**
  - Sample $\mathcal{D}_i = (\boldsymbol{x}_i, \boldsymbol{y}_i)$, $\mathcal{D}_i^* = (\boldsymbol{x}_i^*, \boldsymbol{y}_i^*)$ from $\mathcal{T}_i$;
  - Encode observation $\boldsymbol{z}_i = g_\theta(\boldsymbol{x}_i)$, $\boldsymbol{z}_i^* = g_\theta(\boldsymbol{x}_i^*)$;
  - Compute task specific parameter $\phi_i = \tilde{\theta} + \alpha' \nabla_{\tilde{\theta}} \log q(\boldsymbol{y}_i | \boldsymbol{z}_i, \tilde{\theta})$;
- Update $\tilde{\theta} \leftarrow \tilde{\theta} + \alpha \nabla_{\tilde{\theta}} \sum_{\mathcal{T}_i} \log q(\boldsymbol{y}_i^* | \boldsymbol{z}_i^*, \phi_i)$;
- Update $\tau \leftarrow \tau + \alpha \nabla_\tau [\sum_{\mathcal{T}_i} \log q(\boldsymbol{y}_i^* | \boldsymbol{z}_i^*, \phi_i) - \beta D_{\mathrm{KL}}(q(\theta;\tau) || r(\theta))]$

[4] Yin et al. (2019)
[6] Finn et al. (2017)

B. Meta-Regularised MAML: Algorithm[4] II

Algorithm 3: Meta-Regularized Methods in Meta-testing

input : Meta-testing task $\mathcal{T}$ with training data $\mathcal{D} = (\boldsymbol{x}, \boldsymbol{y})$ and testing input $\boldsymbol{x}^*$, optimized parameters $\tau, \tilde{\theta}$.

output: Prediction $\hat{y}^*$

for *k from 1 to K* **do**

- Sample $\theta_k \sim q(\theta; \tau)$;
- Encode observation $\boldsymbol{z}_k = g_{\theta_k}(\boldsymbol{x})$, $\boldsymbol{z}_k^* = g_{\theta_k}(\boldsymbol{x}^*)$;
- Compute task specific parameter $\phi_k = a(h_{\tilde{\theta}}(\boldsymbol{z}_k, \boldsymbol{y}))$ for MR-CNP and $\phi_k = \tilde{\theta} + \alpha' \nabla_{\tilde{\theta}} \log q(\boldsymbol{y}|\boldsymbol{z}_k, \tilde{\theta})$ for MR-MAML;
- Predict $\hat{y}_k^* \sim q(\hat{y}^*|z_k^*, \phi_k, \tilde{\theta})$

Return prediction $\hat{y}^* = \frac{1}{K}\sum_{k=1}^{K} \hat{y}_k^*$

[4] Yin et al. (2019)

C. Meta-World: Parametric and Non-Parametric Variability

- Without parametric variation, the model could for example memorise that any object at a particular location is a door, while any object at another location is a drawer.
- Position randomisation forces the model to generalise more broadly.
- For example, closing a drawer and pushing a block can appear as nearly the same task for some initial and goal positions of each object.
- Shared underlying structure: The 50 environments require the same robotic arm to interact with different objects, with different shapes, joints, and connectivity.
- The tasks themselves require the robot to execute a combination of reaching, pushing, and grasping, depending on the task.

C. Meta-World: Multi-Task RL vs Meta-Learning RL[5]

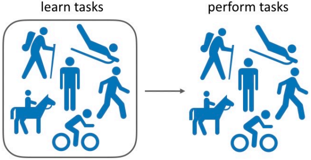


Figure: Multi-task reinforcement learning

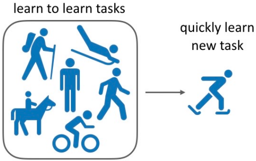


Figure: Multi-task reinforcement learning

- In multi-task RL, we assume that we want to learn a fixed set of skills with minimal data.
- In meta-learning RL, we want to use experience from a set of skills such that we can learn to solve new skills quickly.

[5] Yu et al. (2019)

C. Meta-World: Parametric and Non-Parametric Variability[5]

Figure: Visualisation of two meta-learning evaluation protocols, ranging from within task adaption in ML1, to adapting to new tasks in ML10.

[5] Yu et al. (2019)

D. Meta-Gradient RL[8]

Definition (n-step return)

$$g_\eta(\tau_t) = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ... + \gamma^{n-1} R_{t+n} + \gamma^n v_\theta(S_{t+n}),$$

where $\eta = \{\gamma, \eta\}$.

Definition (λ-return)

The λ-return is a geometric mixture of n-step returns,

$$g_\eta(\tau_t) = R_{t+1} + \gamma(1-\lambda)v_\theta(S_{t+1}) + \gamma\lambda g_\eta(\tau_{t+1}),$$

where $\eta = \{\gamma, \eta\}$.

[8] Xu et al. (2018)

D. Squared Error (prediction update function)[8]

*Definition of the **square error semi-gradient*** derived w.r.t. θ:

$$f(\tau, \theta, \eta) = \alpha(g_\eta(\tau) - v_\theta(s))\frac{\partial v_\theta(s)}{\partial \theta}.$$

[8] Xu et al. (2018)

D. A2C Objective (control update function)[8]

Definition of the ***A2C objective semi-gradient*** derived w.r.t. θ:

$$f(\tau,\theta,\eta) = \alpha[\text{control objective + prediction objective + regulariser}]$$
$$= \alpha\left[(g_\eta(\tau) - v_\theta(s))\frac{\partial \log(\pi_\theta(a|s))}{\partial\theta} + b(g_\eta(\tau) - v_\theta(s))\frac{\partial v_\theta(s)}{\partial\theta} + c\frac{\partial H(\pi_\theta(\cdot|s))}{\partial\theta}\right],$$

where the third term regularises the policy according to its entropy $H(\pi_\theta)$.

[8] Xu et al. (2018)

D. Meta-Objective[8]

(i) *Definition of the **mean square error (prediction) meta-objective:***

$$J'(\tau', \theta', \eta') = (g_{\eta'}(\tau') = v_{\theta'}(s'))^2.$$

(ii) *Definition of the **policy (control) meta-objective:***

$$J'(\tau', \theta', \eta') = (g_{\eta'}(\tau') = v_{\theta'}(s')) \log(\pi_{\theta'}(a'|s')).$$

[8] Xu et al. (2018)

D. Update Meta-Parameters (prediction)[8]

(i) *Definition of the* ***MSE meta-objective semi-gradient:***

$$\frac{\partial}{\partial \theta'} J'(\tau', \theta', \eta') = -2(g_{\eta'}(\tau') - v_{\theta'}(s')) \frac{\partial v_{\theta'}(s')}{\partial \theta'}.$$

(ii) ***Update Meta-Parameters*** η *(prediction):*

$$\begin{aligned} \eta_T &\approx \eta_t - \beta \frac{\partial}{\partial \theta'} J'(\tau', \theta', \eta') \left(\mu \frac{\partial \theta}{\partial \eta} + \frac{\partial}{\partial \eta} f(\tau, \theta, \eta) \right) \\ &\approx \eta_t + 2\beta (g_{\eta'}(\tau') - v_{\theta'}(s')) \frac{\partial v_{\theta'}(s')}{\partial \theta'} \left(\mu \frac{\partial \theta}{\partial \eta} + \frac{\partial}{\partial \eta} f(\tau, \theta, \eta) \right). \end{aligned}$$

[8] Xu et al. (2018)

D. Update Meta-Parameters (control)[8]

(i) *Definition of the **policy gradient objective:***

$$\frac{\partial}{\partial \theta'} J'(\tau', \theta', \eta') = (g_{\eta'}(\tau') - v_{\theta'}(s')) \frac{\partial \log(\pi_{\theta'}(a'|s'))}{\partial \theta'}.$$

(ii) ***Update Meta-Parameters*** η *(control):*

$$\begin{aligned} \eta_T &\approx \eta_t - \beta \frac{\partial}{\partial \theta'} J'(\tau', \theta', \eta') \left(\mu \frac{\partial \theta}{\partial \eta} + \frac{\partial}{\partial \eta} f(\tau, \theta, \eta) \right) \\ &\approx \eta_t - \beta (g_{\eta'}(\tau') - v_{\theta'}(s')) \frac{\partial \log(\pi_{\theta'}(a'|s'))}{\partial \theta'} \left(\mu \frac{\partial \theta}{\partial \eta} + \frac{\partial}{\partial \eta} f(\tau, \theta, \eta) \right). \end{aligned}$$

[8] Xu et al. (2018)

E. Glossar I

Policy optimisation categories ([19] Schulman et al. (2017))

(i) ***Policy iteration methods***, which alternate between estimating the value function under the current policy and improving the policy.

(ii) ***Policy gradient methods***, which use an estimator of the gradient of the expected return (total reward) obtained from sample trajectories.

(iii) ***Derivative-free optimisation methods***, such as the cross-entropy method (CEM) and covariance matrix adaptation (CMA), which treat the return as a black box function to be optimised in terms of the policy parameters.

E. Glossar II

Actor-Critic

The actor *takes as input the state and outputs the best action.* It essentially controls how the agent behaves by ***learning the optimal policy*** (policy-based). ***The critic***, on the other hand, ***evaluates the action by computing the value function*** (value based). Those two models participate in a game where they both get better in their own role as the time passes. The result is that the overall architecture will learn to play the game more efficiently than the two methods separately.

Bootstrapping Update value estimates based on other value estimates.

E. Glossar III

Transfer Learning vs Meta-Learning

Meta-learning *is more about speeding up and optimising hyperparameters for networks that are not trained at all*, whereas ***transfer learning*** *uses a net that has already been trained for some task and reusing part or all of that network to train on a new task which is relatively similar.*

So, although they can both be used from task to task to a certain degree, they are completely different from one another in practice and application, *one tries to optimise configurations for a model and the other simply reuses an already optimised model, or part of it at least.*

E. Glossar IV

Human starts

The first protocol is "human starts" which initialises episodes to a state that is randomly sampled from human play.

No-op starts

"No-ops starts" initialise each episode with a random sequence of no-op actions; this protocol is also used during training in the meta-gradient RL paper.