Meta-Learning across Time

Chelsea Finn
Learning from scratch is a **limitation**, not a benefit.

Why is it hard to reuse prior experience?
- what to transfer?
- what **is there** to transfer?

How can we **prepare** for the future while learning?

Optimize for **transferability** to new tasks, i.e. learn to learn efficiently
Outline

1. How to meta-learn?
2. What tasks do we meta-learn across?
3. Can we learn a prior effective for continual learning?
Outline

1. How to meta-learn?
2. What tasks do we meta-learn across?
3. Can we learn a prior effective for continual learning?
Few-Shot Learning Example

Given 1 example of 5 classes:

![Images of example classes](image1) ![Images of example classes](image2) ![Images of example classes](image3) ![Images of example classes](image4) ![Images of example classes](image5)

Classify new examples

training data $D_{\text{train}}$

![Image of training data](image6)

test set $X_{\text{test}}$

meta-training

$\mathcal{T}_1$

$\mathcal{T}_2$

training classes

One slice of the continual learning problem

diagram adapted from Ravi & Larochelle '17
Design of learner?

- Continue to learn from more data?

Santoro et al. ‘16, Duan et al. ‘17, Wang et al. ‘17, Munkhdalai & Yu ‘17, Mishra et al. ‘17, …

Snell et al. ‘17
Vinyals et al. ‘16
Hochreiter et al. ‘01
Andrychowicz et al. ‘16
Li & Malik ‘16

and many many more approaches

Recurrent network
(LSTM, NTM, Conv)

\[
y_{\text{test}} = f(D_{\text{train}}, x_{\text{test}}; \theta)
\]

Santoro et al. ‘16, Duan et al. ‘17, Wang et al. ‘17, Munkhdalai & Yu ‘17, Mishra et al. ‘17, …

+ expressive, general
+ applicable to range of problems
- continue to learn from more data?
Algorithm more suitable for continual learning?

Fine-tuning
[test-time]

Our method

\[
\min_{\theta} \sum_{\text{task } i} L^i_{\text{test}} (\theta - \alpha \nabla_\theta L^i_{\text{train}}(\theta))
\]

Key idea: Train over many tasks, to learn parameter vector \( \theta \) that transfers

**Model-Agnostic Meta-Learning**

Design of $f$?

Recurrent network

$$y_{test} = f(D_{train}, x_{test}; \theta)$$

network implements the "learned learning procedure"

What to do if not good enough?
- Nothing

MAML

$$y_{test} = f(x_{test}; \theta - \alpha \nabla_{\theta} \mathcal{L}(D_{train}))$$

What to do if not good enough?
- Keep taking gradient steps (it's gradient descent..)
Where should the meta-learning tasks come from?

Can we instead meta-learn across time?
Can we instead meta-learn across time?

- gradual terrain change
- motor malfunction

Nagabandi*, Clavera*, Liu, Fearing, Abbeel, Levine, Finn. Learning to Adapt in Dynamic Real-World Environments through Meta-RL
Can we instead meta-learn across time?

tasks are windows of experience
**tasks** are **windows** of experience

**Goal:** perform well at the current timestep

Use your **recent experience** to inform current timestep

\[
\min_{\theta} \sum_{\text{task } t} \mathcal{L}_{\text{test}}^t (\theta - \alpha \nabla_{\theta} \mathcal{L}_{\text{train}}^t (\theta)) \quad (\text{applicable to other meta-learning algorithms too})
\]

**Assumptions & Requirements**
- assume [replay] buffer of previous experience
- need ability to learn off-policy
Model-based Reinforcement Learning

Collect data

Fit model

Plan using the model

Model-based RL achieves zero-shot generalization to new rewards*

Can we continually learn & adapt to new dynamics/environments?

Meta-training: \[
\min_{\theta} \sum_{\text{task } t} \mathcal{L}_{\text{test}}^t (\theta - \alpha \nabla_{\theta} \mathcal{L}_{\text{train}}^t (\theta))
\]

*with some caveats
Meta-training:

$$\min_{\theta} \sum_{\text{task } t} \mathcal{L}_{\text{test}}^t(\theta - \alpha \nabla_{\theta} \mathcal{L}_{\text{train}}^t(\theta))$$

- model error on $s_{t:t+H}, a_{t:t+H-1}$
- model error on $s_{t-M:t}, a_{t-M:t-1}$

Result: meta-learned initialization $\theta$

Test time:
At every time step $t$:

1. Reset to learned initialization $\theta$
2. Adapt model parameters $\theta' = \theta - \alpha \nabla_{\theta} \mathcal{L}_{\text{train}}^t(\theta)$
3. Plan using adapted model
Dynamic Environments **without** Adaptation

Model-Based RL Only
Dynamic Environments with Online Adaptation

with MAML

Nagabandi*, Clavera*, Liu, Fearing, Abbeel, Levine, Finn. Learning to Adapt in Dynamic Real-World Environments through Meta-RL
VelociRoACH Robot

Meta-train on variable terrains

Meta-test with slope, missing leg, payload, calibration errors

Nagabandi*, Clavera*, Liu, Fearing, Abbeel, Levine, Finn. Learning to Adapt in Dynamic Real-World Environments through Meta-RL
Roach Robot

**Meta-train on variable terrains**  **Meta-test with slope, missing leg, payload, calibration errors**

<table>
<thead>
<tr>
<th>Return</th>
<th>GrBAL (ours)</th>
<th>MB (no metalearning, no adaptation)</th>
<th>MB+DE (no metalearning)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slope/Terrain</td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
</tr>
<tr>
<td>Missing Leg</td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
</tr>
<tr>
<td>Payload (Straight)</td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
</tr>
<tr>
<td>Payload (Right)</td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
</tr>
<tr>
<td>Pose shift</td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
<td><img src="graph.png" alt="Graph showing performance comparison" /></td>
</tr>
</tbody>
</table>

Nagabandi*, Clavera*, Liu, Fearing, Abbeel, Levine, Finn. Learning to Adapt in Dynamic Real-World Environments through Meta-RL
Meta-train on variable terrains  Meta-test with **slope**, missing leg, payload, calibration errors

Roach Robot

GrBAL (ours)  Model-Based RL (no adaptation)

Nagabandi*, Clavera*, Liu, Fearing, Abbeel, Levine, Finn. Learning to Adapt in Dynamic Real-World Environments through Meta-RL
Roach Robot

Meta-train on variable terrains  Meta-test with slope, **missing leg**, payload, calibration errors

GrBAL (ours)  Model-Based RL (no adaptation)

Nagabandi*, Clavera*, Liu, Fearing, Abbeel, Levine, Finn. Learning to Adapt in Dynamic Real-World Environments through Meta-RL
Outline

1. How to meta-learn?
2. What tasks do we meta-learn across?
3. Can we learn a prior effective for continual learning?
gradual terrain change

motor malfunction

time
M time steps not sufficient to learn entirely new terrain

Continue to run gradient descent?

Need to recognize similarities & differences across time
Online inference problem: infer latent “task” variable at each time step

Mixture of neural networks over task variable T, adapted continually: $\theta_t(T_i)$

Alternate between:

E-step: Estimate latent “task” variable at each time step $P(T_t)$ given data $x_t, y_t$

$$P(T_t = T_i | x_t, y_t) \propto p_{\theta(T_i)}(y_t | x_t, T_t = T_i) P(T_t = T_i)$$

likelihood of the data under task $T_i$ prior

M-step: Update mixture of network parameters

$$\theta_{t+1}(T_i) = \theta_t(T_i) - \beta P(T_t = T_i | x_t, y_t) \nabla_{\theta_t(T_i)} \log p_{\theta_t(T_i)}(y_t | x_t) \quad \forall T_i$$

gradient step on each mixture element, weighted by task probability

Note: Online learning with neural nets won’t work in the general case.

Nagabandi, Finn, Levine. Deep Online Learning via Meta-Learning
Does it work?

Crawler with crippled legs

Constant crippled vs. crippled vs. crippled

Normalized Reward

Constant crippled setting

Regions of normal/crippled leg

Preparing for the future is critical

Nagabandi, Finn, Levine. Deep Online Learning via Meta-Learning
Does it work?

Crawler with crippled legs

Next step:

Latent task distribution during online learning

Nagabandi, Finn, Levine. Deep Online Learning via Meta-Learning
Takeaways

Temporal windows is a general mechanism for task distributions.

Meta-learning provides an effective prior for future learning.

Future Work

Continual meta-learning

Stream of non-stationary dynamics and rewards

Collaborators

Anusha Nagabandi  Ignasi Clavera  Simin Liu  Pieter Abbeel  Ron Fearing  Sergey Levine

Papers, data, and code linked at: people.eecs.berkeley.edu/~cbfinn

Questions?