Flexible Neural Networks and the Frontiers of Meta-Learning

Chelsea Finn
How can we enable agents to learn skills in the real world?

Why robots?

Robots can teach us things about intelligence.

faced with the real world

must generalize across tasks, objects, environments, etc

need some common sense understanding to do well

supervision can’t be taken for granted
Learn one task in one environment, starting from scratch, with detailed supervision.

Not just a problem with reinforcement learning & robotics.

More diverse, yet still one task, from scratch, with detailed supervision.
Can we enable systems to **accumulate experiences** and **acquire general-purpose priors** that enable fast learning and reasoning?
By Braque or Cezanne?
How did you accomplish this?

Through previous experience.
How should we incorporate prior experience into ML systems?

- Modeling image formation
- Geometry
- SIFT features, HOG features + SVM
- Fine-tuning from ImageNet features
- Domain adaptation from other painters

Can we explicitly learn priors from previous experience that lead to efficient downstream learning?

Can we learn to learn?
How should we incorporate *prior experience* into ML systems?

First: a primer on meta-learning & what it can accomplish

Second: challenges & frontiers
Example: Few-Shot Image Classification

5-way, 1-shot image classification \((\text{MinilImagenet})\)

Given 1 example of 5 classes:

Classify new examples

Can replace image classification with: regression, language generation, skill learning, any ML problem.
The Meta-Learning Problem: The Mechanistic View

Supervised Learning:
- Inputs: \( x \)
- Outputs: \( y \)
- Data: \( \{(x, y)_i\} \)

\[
y = f(x; \theta)
\]

Meta-Supervised Learning:
- Inputs: \( D_{\text{tr}} \)
- Outputs: \( y_{\text{test}} \)
- Data: \( \{D_i\} \)

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

Why is this view useful?
Reduces the problem to the design & optimization of \( f \).

Finn. Learning to Learn with Gradients. PhD thesis 2018
The Meta-Learning Problem: The **Probabilistic View**

**Supervised Learning:**
Inputs: $\mathbf{x}$  
Outputs: $\mathbf{y}$  
Data: $\{(\mathbf{x}, \mathbf{y})_i\}$

$\mathbf{y} = f(\mathbf{x}; \theta)$

As inference: $p(\theta|\mathcal{D})$

**Meta-Supervised Learning:**
Inputs: $\mathcal{D}_{\text{tr}}$  
Outputs: $\mathbf{y}_{\text{test}}$  
Data: $\{\mathcal{D}_i\}$

$\mathbf{y}_{\text{test}} = f(\mathcal{D}_{\text{tr}}, \mathbf{x}_{\text{test}}; \theta)$

As inference: $p(\phi_i|\mathcal{D}_{\text{tr}}^i, \theta) \quad \max_{\theta} \sum_i \log p(\phi_i|\mathcal{D}_{\text{ts}}^i)$
Few-Shot Learning

- expressive, general
- applicable to range of problems
- complex model for complex task of learning
- often large data requirements

Recurrent network
(LSTM, NTM, Conv)

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

Santoro et al. ‘16, Duan et al. ’17, Wang et al. ’17, Munkhdalai & Yu ’17, Mishra et al. ‘17, …
Optimization-Based Inference

Meta-learning

\[
\min_{\theta} \sum_{\text{task } i} L(\theta - \alpha \nabla_{\theta} L(\theta, D_{i}^{tr}), D_{i}^{ts})
\]

Key idea: Over many tasks, learn parameter vector $\theta$ that transfers via fine-tuning

Fine-tuning

\[
\phi \leftarrow \theta - \alpha \nabla_{\theta} L(\theta, D^{tr})
\]
Optimization-Based Inference

$$\min_{\theta} \sum_{\text{task } i} \mathcal{L}(\theta - \alpha \nabla_\theta \mathcal{L}(\theta, \mathcal{D}_{i}^{\text{tr}}), \mathcal{D}_{i}^{\text{ts}})$$

$\theta$ parameter vector being meta-learned

$\phi_i^*$ optimal parameter vector for task $i$

---

Model-Agnostic Meta-Learning

To give some intuition…

**two tasks**: running backward, running forward

Can we learn a representation under which RL is fast and efficient?

What can we do with meta-learning?
Leverage data with previous **environments** to quickly adapt to new ones?
Leverage data with previous **objects** to quickly adapt to new ones?

Previous demo data from *training objects*

subset of training objects
tasks: placing object into target container

**input demo** (via teleoperation)

**resulting policy**

Leverage data with previous objects to quickly adapt to new ones?

Previous demo data + human video data from training objects

subset of training objects

tasks: placing object into target container

input *human* demo

resulting policy

Yu*, Finn*, Xie, Dasari, Abbeel, Levine. One-Shot Imitation from Observing Humans via Domain-Adaptive Meta-Learning. RSS 2018
Leverage previous **language** experience

Low-resource neural machine translation

Leverage previous **segmentation** experience

Image segmentation from a few pixel labels

Leverage previous experience with **domains**

Leverage experience with previous **domains**

see, e.g.: Gu et al. EMNLP ’18

Leverage previous experience with **objects**

Few-shot image generation

see, e.g.: Shaban, et al. One-Shot Learning for Semantic Segmentation.
Dong & Xing. Few-Shot Semantic Segmentation with Prototype Learning.

Leverage previous experience with **people**

Personalize dialog to a persona

see, e.g.: Lin*, Madotto* et al. ACL ’19

see, e.g.: Li et al. Learning to Generalize: Meta-Learning for Domain Adaptation.

see, e.g.: Gordon et al. VERSA: Versatile and Efficient Few-Shot Learning.

And many many others…
How should we incorporate *prior experience* into ML systems?

The algorithms work pretty well.

But is the problem statement what we want?

Prior experience doesn’t typically come all at once. What are the tasks and where do they come from?
Meta-Learning
(Schmidhuber et al. ’87, Bengio et al. ’92)
Given i.i.d. task distribution,
learn a new task efficiently

More realistically:
slow learning  rapid learning

learn  learn  learn  learn  learn  learn  learn  learn
**Meta-Learning**  
(Schmidhuber et al. ’87, Bengio et al. ’92)  
Given i.i.d. task distribution, learn a new task efficiently

**Online Learning**  
(Hannan ’57, Zinkevich ’03)  
Perform sequence of tasks while minimizing static regret.

**Online Meta-Learning**  
(this work)  
Efficiently learn a sequence of tasks from a non-stationary distribution.

---

The Online Meta-Learning Problem Setting

For round $t \in \{1, 2, \ldots, \infty\}$:

1. World picks a loss function $l_t(\cdot)$
2. Agent should pick $\theta_t$ without knowledge of $l_t$
3. Agent uses update procedure $\Phi_t : \Theta \rightarrow \Theta$, and obtains $\tilde{\theta}_t = \Phi_t(\theta_t)$
4. Agent suffers $l_t(\tilde{\theta}_t)$ for the round

**Goal:** Learning algorithm with sub-linear regret

$$\text{Regret}_T := \sum_{t=1}^{T} l_t(\Phi_t(\theta_t)) - \min_{\theta \in \Theta} \sum_{t=1}^{T} l_t(\Phi_t(\theta))$$
Follow the Meta-Leader (FTML): \[
\theta_{t+1} = \arg \min_\theta \sum_{t=1}^{T} \ell_t(\Phi_t(\theta))
\]

Can be implemented with MAML

**Theorem** (Informal): If \(\{\ell_t(\cdot), \hat{\ell}_t(\cdot)\} \ \forall t\) are \(C^2\)-smooth and strongly convex, the sequence of models \(\{\theta_1, \theta_2, \ldots, \theta_T\}\) returned by FTML has the property:

\[
\text{Regret}_T := \sum_{t=1}^{T} \ell_t(\Phi_t(\theta_t)) - \min_{\theta \in \Theta} \sum_{t=1}^{T} \ell_t(\Phi_t(\theta)) = O(\log T)
\]
Practical instantiation of FTML: meta-train with MAML on all data so far, fine-tune on current task

Experiment with sequences of tasks:
- Colored, rotated, scaled MNIST
- 3D object pose prediction
- CIFAR-100 classification

Compare to:
- TOE (train on everything): train on all data so far
- FTL (follow the leader): train on all data so far, fine-tune on current task
- From Scratch: train from scratch on each task

Example pose prediction tasks:
- plane
- car
- chair

Experiments

TOE (train on everything): improves over time, but **prone to negative transfer**

FTL (follow the leader): **consistent forward transfer**, sometimes **overfits**

FTML (ours): **learns each new task faster** & **with greater proficiency**, approaches **few-shot learning regime**
How should we incorporate prior experience into ML systems?

The algorithms work pretty well.

But is the problem statement what we want?

Prior experience doesn’t typically come all at once. What are the tasks and where do they come from?
Meta-learning: manual algorithm design —> manual task distribution design

Can we also automate the task design process?
Propose tasks for meta-learning with only **unlabeled** images?

Construct tasks without labeled data?

Unsupervised learning (to get an embedding space) → Propose tasks → Run meta-learning

Result: representation suitable for learning downstream tasks

Hsu, Levine, Finn. Unsupervised Learning via Meta-Learning. ICLR’19
Propose tasks for meta-learning with only **unlabeled** images?

Unsupervised learning (to get an embedding space) → Propose tasks → Run meta-learning

A few options:
- BiGAN — Donahue et al. ‘17
- DeepCluster — Caron et al. ‘18

Clustering to Automatically Construct Tasks for Unsupervised Meta-Learning (CACTUs)

Method | Accuracy
--- | ---
MAML with labels | 62.13%
BiGAN kNN | 31.10%
BiGAN logistic | 33.91%
BiGAN MLP + dropout | 29.06%
BiGAN cluster matching | 29.49%
BiGAN CACTUs MAML | 51.28%
DeepCluster CACTUs MAML | 53.97%

**Same story for:**
- 4 different embedding methods
- 4 datasets (Omniglot, CelebA, miniImageNet, MNIST)
- 2 meta-learning methods (*)
- Test tasks with larger datasets

*ProtoNets underperforms in some cases.

Hsu, Levine, Finn. Unsupervised Learning via Meta-Learning. ICLR’19
What about unsupervised meta-RL?

Environment → Propose tasks → Run meta-RL

Result: Environment-specific RL algorithm

What about unsupervised meta-RL?

Environment ➔ Propose tasks ➔ Run meta-RL

- Propose tasks using skill discovery methods (e.g. DIAYN)

  latent skill: \( z \)  
  policy: \( \pi(a|s, z) \)  
  discriminator: \( D(z|s) \)

  (discrete latent variable)

**Goal:** Maximize *mutual information* between \( s, z \)

- Policy \( \rightarrow \) visit states that are discriminable
- Discriminator \( \rightarrow \) predict skill from state

Examples of acquired tasks:

Task reward for meta-learning:

\[
    r(s, z) = \log D(z|s)
\]
Does it work?

Measure learning performance on test tasks with rewards

Cheetah

Ant

Takeaway: Relatively simple mechanisms for proposing tasks work surprisingly well.

What about unsupervised meta-RL?

Environment $\rightarrow$ Propose tasks $\rightarrow$ Run meta-RL

Can we adapt the task distribution based on the meta-learner’s current behavior?

Formulate task acquisition as an information maximization problem, optimized with EM.

Learn latent representation $\mathbf{s}$ through deep trajectory-centric clustering.

Fit generative mixture model over $\mathbf{s}$.

Meta-train w.r.t. mutual information objective under density model.

Jabri, Hsu, Eysenbach, Gupta, Efros, Levine, Finn. Unsupervised Curricula for Visual Meta-RL. ’19
Natural to incorporate **density-based exploration**.

\[
    r_z(s) = \lambda \log q_{\phi}(s|z) - \log q_{\phi}(s)
\]

**Meta-train**

acquire skills and explore

Scales naturally to **visual observations**.

Directly transfers to **downstream** tasks.

VizDoom

(a) ViZDoom (fixed)  (b) ViZDoom (random)  (c) Sawyer

+ improves with further EM steps

Jabri, Hsu, Eysenbach, Gupta, Efros, Levine, Finn. Unsupervised Curricula for Visual Meta-RL. ’19
How should we incorporate prior experience into ML systems?

Meta-learning provides a way to optimize for priors that lead to few-shot learning.

Prior experience doesn’t typically come all at once.

> **online meta-learning** setting

What are the tasks and where do they come from?

> can propose tasks from unlabeled experience

Both of these bring us closer towards realistic lifelong learning scenarios.
Can we perform meta-RL across **distinct task families**?

**Meta-World benchmark**

- 50 distinct control tasks
- shared workspace, action-space
- designed for studying **multi-task** transfer, meta-learning

T Yu, D Quillen, Z He, R Julian, K Hausman, S Levine, C Finn
Collaborators & Students

Sergey Levine  Tianhe Yu  Anusha Nagabandi  Kyle Hsu  Ignasi Clavera  Pieter Abbeel  Kate Rakelly  Aurick Zhou  Russell Mendonca

Annie Xie  Sudeep Dasari  Simin Liu  Avi Singh  Abhishek Gupta  Ben Eysenbach  Allan Jabri

Questions?