Lifelong Learning

CS 330

Course Reminders

Optional homework 4 due **today**.

Project milestone due **Wednesday.**

Guest lecture on Wednesday!

Hanie Sedghi

Please try to show up in person & on-time.

Plan for Today

The lifelong learning **problem statement**

Basic approaches to lifelong learning

Can we do **better** than the basics?

Revisiting the problem statement
from **the meta-learning perspective**

A brief review of problem statements.

Multi-Task Learning

Learn to solve a set of tasks.

learn tasks

perform tasks

Meta-Learning

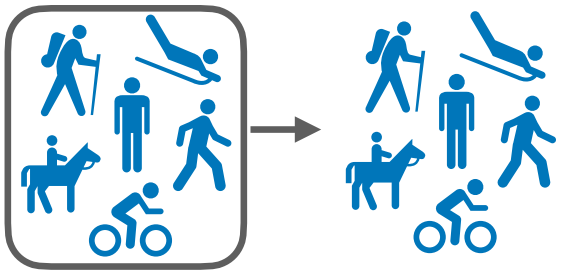
Given i.i.d. task distribution,
learn a new task efficiently

learn to learn tasks

quickly learn
new task

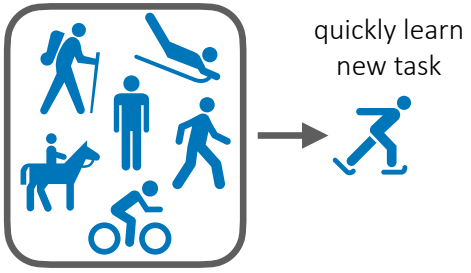
Multi-Task Learning

learn tasks → perform tasks



Meta-Learning

learn to learn tasks → quickly learn new task



In contrast, many real world settings look like:

Our agents may not be given a large batch of data/tasks right off the bat!

Some examples:

→ curriculum that increases in difficulty, ~ subsequent tasks build on previous tasks

- a **student** learning concepts in school
- a deployed **image classification system** learning from a stream of images from users
- a **robot** acquiring an increasingly large set of skills in different environments
- a **virtual assistant** learning to help different users with different tasks at different points in time
- a **doctor's assistant** aiding in medical decision-making

Some Terminology

Mostly Synonyms

Sequential learning settings

↳ online learning, lifelong learning, continual learning, incremental learning, streaming data

≠ distinct from sequence data and sequential decision-making

↳ Data points have a sequential structure

↳ system is making multiple decisions in sequence

≠

Learning as the data comes in

(sequential (continual / lifelong learning)

What is the lifelong learning *problem statement*?

Exercise: 1. Pick an example setting.

2. Discuss problem statement in small groups:

(a) how would you set-up an experiment to develop & test your algorithm?

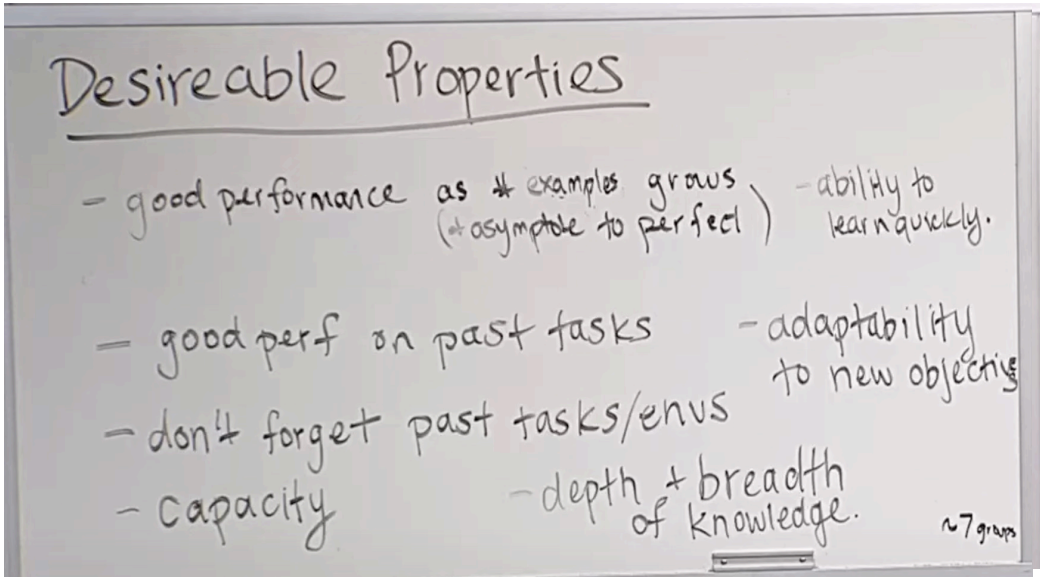
(b) what are desirable/required properties of the algorithm?

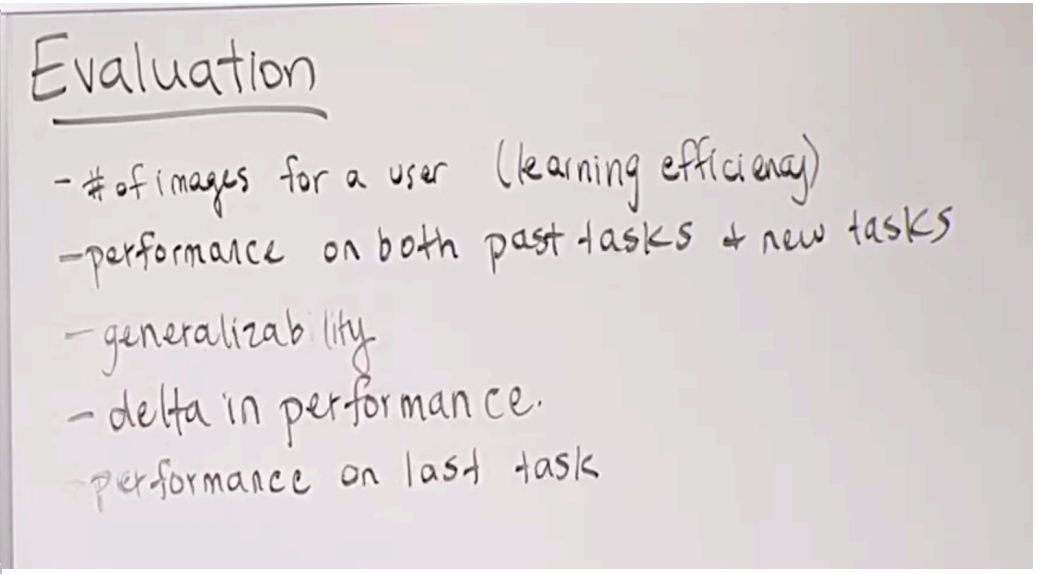
(c) how do you evaluate such a system?

Example settings:

A. a **student** learning concepts in school
B. a deployed **image classification system** learning from a stream of images from users
C. a **robot** acquiring an increasingly large set of skills in different environments
D. a **virtual assistant** learning to help different users with different tasks at different points in time
E. a **doctor's assistant** aiding in medical decision-making

What is the lifelong learning *problem statement*?





What is the lifelong learning *problem statement*?

Problem variations:

- **task/data order**: i.i.d. vs. predictable vs. curriculum vs. adversarial
  - e.g. detect spam over time → spammers might make ways to get thru the filters → Adversarial setting
- **discrete** task boundaries vs. **continuous** shifts (vs. both)
  - discrete → New users
  - continuous → e.g. people's sentiments on certain topics, seasons, etc.
- **known** task boundaries/shifts vs. **unknown**

Some considerations:

- model **performance**
- data **efficiency** → ability to learn tasks efficiently over time
- **computational** resources
- **memory** → may not be able to simultaneously see all accumulated data (history)
  - too big
  - may not be able to store data over a fixed period of time.
  - Data silos
- others: privacy, interpretability, fairness, test time compute & memory

"Differential Privacy"
⇒ Set up the model s.t. you can't get the data back out from the model.

Substantial variety in problem statement!

What is the lifelong learning *problem statement*?

General [supervised] online learning problem:

for t = 1, ..., n

observe x_t <— if **observable task boundaries**: observe x_t, z_t

predict $\hat{y}_t$

observe label y_t

i.i.d. setting: $x_t \sim p(x),\ y_t \sim p(y|x)$

p not a function of t

otherwise: $x_t \sim p_t(x),\ y_t \sim p_t(y|x)$

streaming setting: cannot store (x_t, y_t)

- lack of memory
- lack of computational resources
- privacy considerations
- want to study neural memory mechanisms

true in some cases, but not in many cases!

- recall: replay buffers

What do you want from your lifelong learning algorithm?

Q: Calculate best learner (in hindsight) using the last (best) parameter θ?

A: Note, we can't evaluate regret in practice bec. we don't have access to the best possible params (θ^*). → Approximate θ^* @ the end of training by training a model using all observed data.

Q: Could the regret value < 0 ?

A: Typically in i.i.d. settings, there exists a single model that can do well on all of the data. But in non-i.i.d. setting, you can get a (-)ve regret.

minimal regret (that grows slowly with t)

regret: cumulative loss of learner — cumulative loss of best learner in hindsight

$$\text{Regret}_T := \sum_{1}^{T} \mathscr{L}_t(\theta_t) - \min_{\theta} \sum_{1}^{T} \mathscr{L}_t(\theta)$$

(*) no subscript! → Function of T (capital)

(cannot be evaluated in practice, useful for analysis)

Regret that grows linearly in t is trivial. Why?

↳ Constant θ_t through t will give this result → Models not being updated.

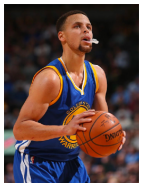
What do you want from your lifelong learning algorithm?

minimal regret (that grows slowly with t)

regret: cumulative loss of learner — cumulative loss of best learner in hindsight

$$\text{Regret}_T := \sum_1^T \mathscr{L}_t(\theta_t) - \min_\theta \sum_1^T \mathscr{L}_t(\theta)$$

1st game 2nd game 3rd game

t	1	2	3
			
$\hat{y}_t$	10 (10)	~~10~~ 30	~~10~~ 29
y_t	30	28	32

Avg. of 30 & 28. ⇒ Avg. of all previous values

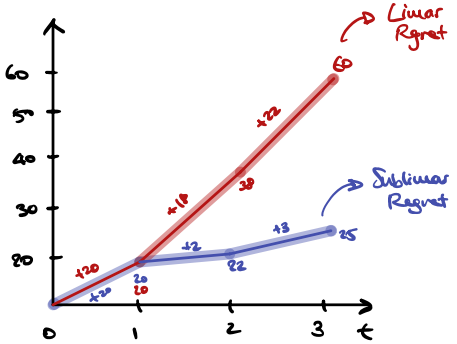
No. of pts. the player will score over time

@ t=T (=3),

20 + 18 + 22

20 + 2 + 3

t	A	B
1	20	20
2	38	22
3	60	25



We want sublinear regret!

⇒ Means we're getting better at making predictions on your online learning prob.

↳ Avg of all prev. values is a better predictor than constant (0) predictor

What do you want from your lifelong learning algorithm?

positive & negative transfer

positive forward transfer: previous tasks cause you to do better on future tasks compared to learning future tasks from scratch

positive backward transfer: current tasks cause you to do better on previous tasks compared to learning past tasks from scratch

eg) For each task you don't have much data (eg 100 data pts/task)
→ Once you accumulate enough data from all the tasks observed so far, you may do better on the 1st task.

positive -> negative : better -> worse

i.e. negative fwd transfer : prev tasks cause you to do worse on future tasks.

etc.

aka "catastrophic forgetting"

Negative Backward Transfer is quite common if you have a small memory (basically start forgetting the past tasks)

Positive Forward Transfer is also common → kind of like pretraining on the previous tasks rather than starting from scratch.

Plan for Today

The lifelong learning **problem statement**

Basic approaches to lifelong learning

Can we do **better** than the basics?

Revisiting the problem statement
from **the meta-learning perspective**

Approaches

@ $t=T$,
$D_T = \bigcup_{t=1}^{T} (x_t, y_t)$
$\min_\theta \mathcal{L}(\theta, D_T)$

Store all the data you've seen so far, and train on it. —> follow the leader algorithm

+ will achieve very strong performance
- computation intensive —> Continuous fine-tuning can help.
- can be memory intensive [depends on the application]

Take a gradient step on the datapoint you observe. —> stochastic gradient descent

@ $t=T$,
$\theta = \theta - \nabla_\theta \mathcal{L}(\theta, \{x_T, y_T\})$ → Update the model using just the dataset retrieved at $t=T$.

+ computationally cheap
+ requires 0 memory
- subject to negative backward transfer → Common when you have small memory
"forgetting"
- slow learning

sometimes referred to as catastrophic forgetting

Can we do better? → The above methods actually work pretty well in a lot of cases

Applying a simple continual learning algorithm to robotics

7 robots collected 580k grasps

86%

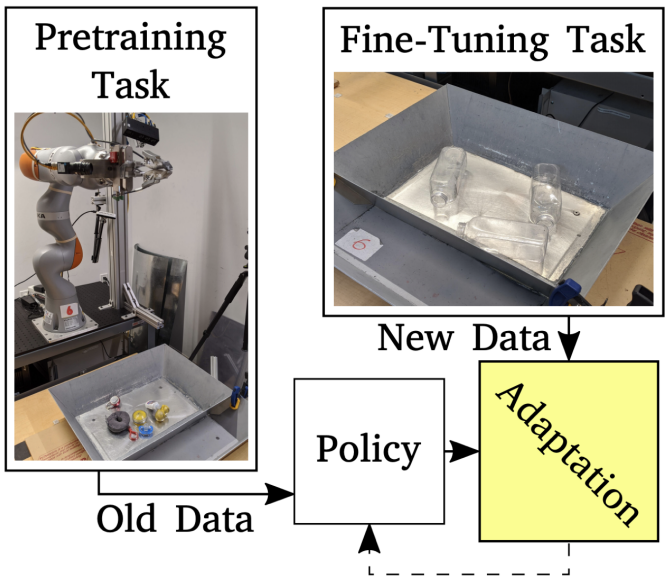
↳ Grasp success rate

49%

↳ Lowered success rate for different-looking (new) objects

Julian, Swanson, Sukhatme, Levine, Finn, Hausman, Never Stop Learning, 2020

Applying a simple continual learning algorithm to robotics

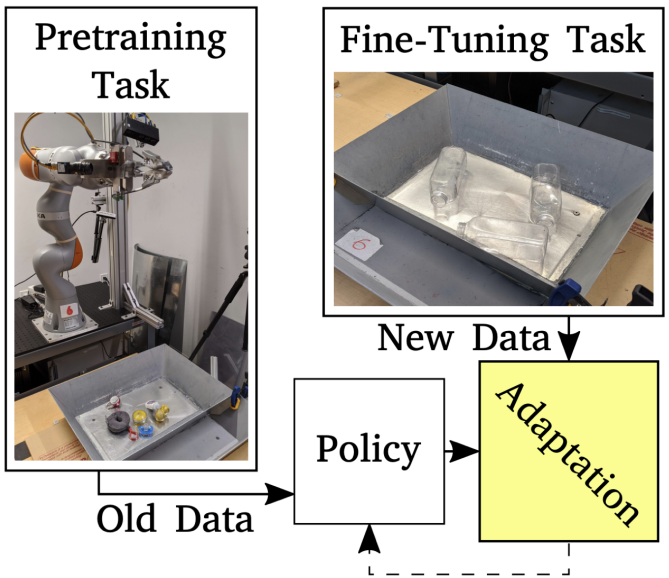


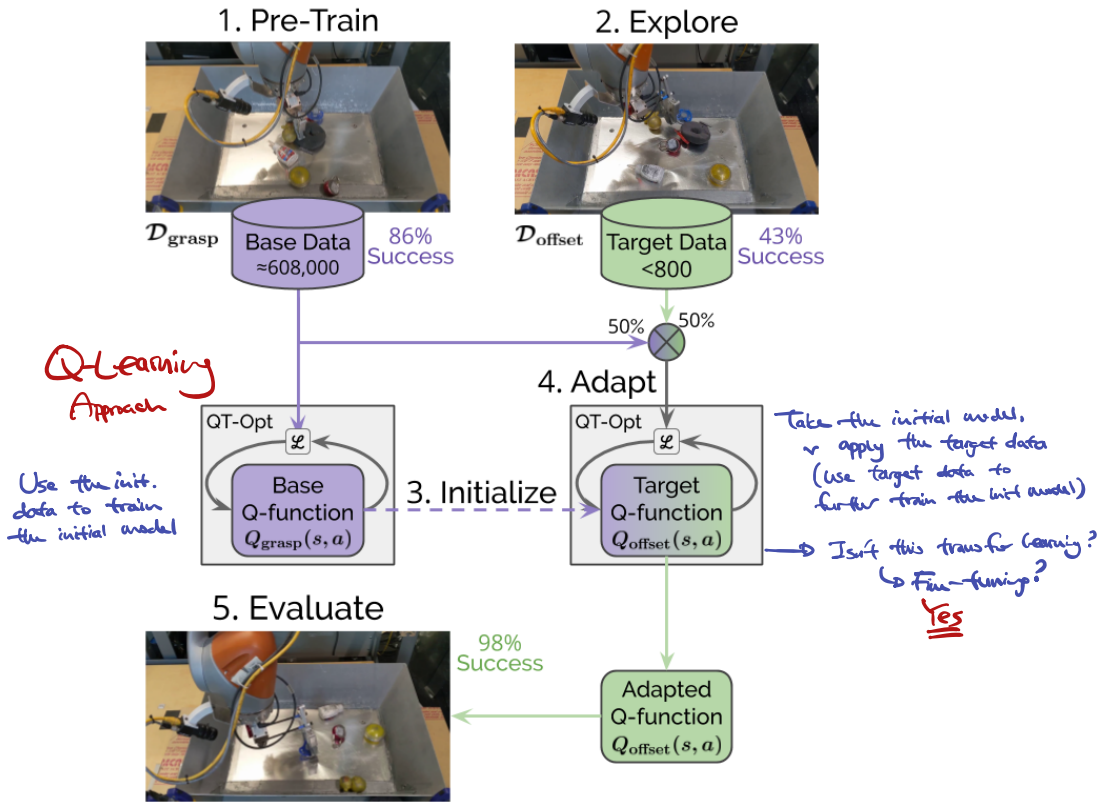
86%

49%

Julian, Swanson, Sukhatme, Levine, Finn, Hausman, Never Stop Learning, 2020

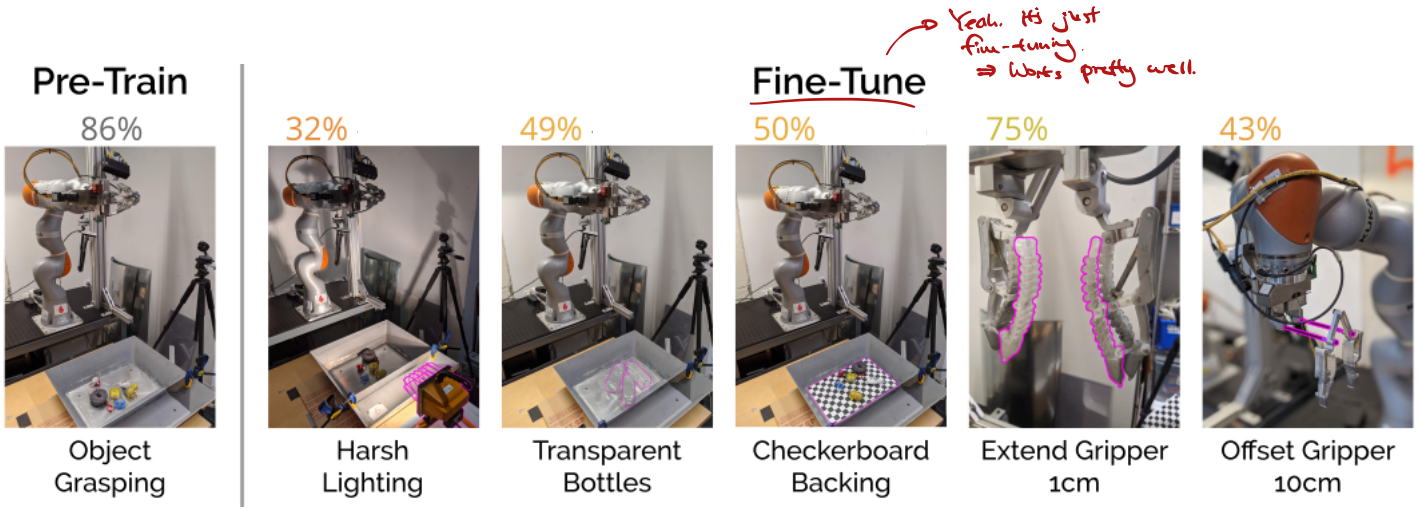
Applying a simple continual learning algorithm to robotics

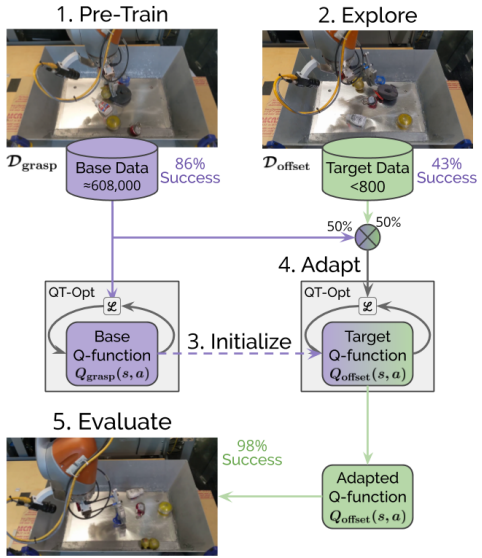




Julian, Swanson, Sukhatme, Levine, Finn, Hausman, Never Stop Learning, 2020

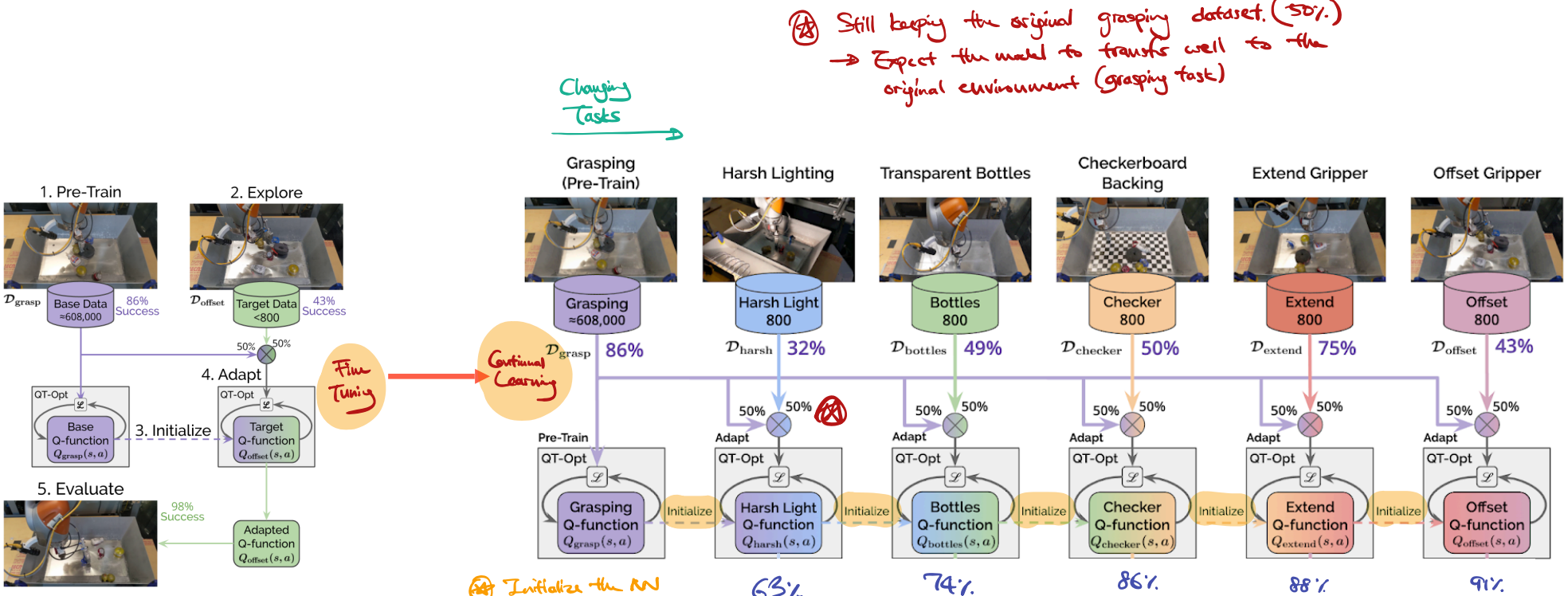
Applying a simple continual learning algorithm to robotics





Julian, Swanson, Sukhatme, Levine, Finn, Hausman, Never Stop Learning, 2020

Applying a simple continual learning algorithm to robotics



What about backward transfer?

↳ aka "catastrophic forget"

Can we do better?

↳ Avoid (-)ve backward transfer while using a small amount of memory

Julian, Swanson, Sukhatme, Levine, Finn, Hausman, Never Stop Learning, 2020

Plan for Today

The lifelong learning **problem statement**

Basic approaches to lifelong learning

Can we do **better** than the basics?

Revisiting the problem statement
from **the meta-learning perspective**

Case Study: Can we modify vanilla SGD to avoid negative backward transfer?
(from scratch)

Idea:

(1) store small amount of data per task in **memory**

(2) when making updates for new tasks, ensure that they don't **unlearn** previous tasks

How do we accomplish (2)?

learning predictor $y_t = f_\theta(x_t, z_t)$ memory: $\mathcal{M}_k$ for task z_k

↳ as few as 5 examples

For $t = 0,\ldots,T$

minimize $\mathcal{L}(\ f_\theta(\cdot, z_t)\ ,\ (x_t, y_t)\ )$

subject to $\mathcal{L}(\ f_\theta\ ,\ \mathcal{M}_k\ ) \leq \mathcal{L}(\ f_\theta^{t-1}\ ,\ \mathcal{M}_k\ )$ for all $k < t$

↳ Using memories of just the previous tasks. (not including current)

(i.e. s.t. loss on previous tasks doesn't get worse)

↳ If you're making a small update to the model, and if it doesn't make the predictions on some of the datasets worse, it probably wouldn't make other things a lot worse either.

Instead of placing a constraint on the loss function directly, place it on the gradients.

Assume local linearity:

If the gradients are pointing in similar directions, you'll get (+)ve backwards transfer

Ensure that the gradient of the func. points in the same direction ~ orthogonal to gradients for the previous task.

↳ Assuming local linearity, → means that the loss for prev. task wasn't increased.

Inner product

$$\langle g_t, g_k \rangle := \left\langle \frac{\partial \mathcal{L}(\ f_\theta\ ,\ (x_t, y_t)\ )}{\partial \theta}, \frac{\partial \mathcal{L}(\ f_\theta\ ,\ \mathcal{M}_k\ )}{\partial \theta} \right\rangle \geq 0 \quad \text{for all } z_k < z_t$$

Quadratic Programming

Can formulate & solve as a QP.

(A) It's possible that this constraint leads to an infeasible constrained optimization. (i.e. current & prev tasks's gradients might be pointing to opposite ways). → may be increasingly infeasible as the no. of past tasks ↑. (∵ increase the no. of constraints on the optimization).

Lopez-Paz & Ranzato. Gradient Episodic Memory for Continual Learning. NeurIPS '17

Experiments

Problems:

- MNIST permutations
- MNIST rotations
- CIFAR-100 (5 new classes/task)

Each task has a diff. permutation of the pixels in the image

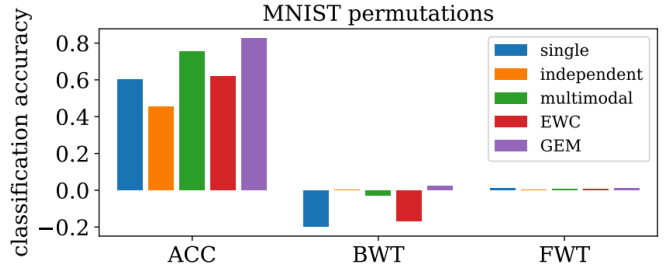
Randomly rearranging the pixels

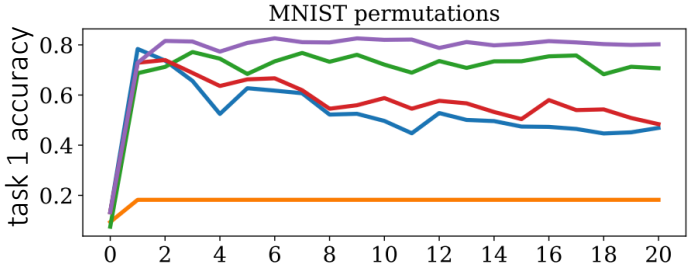
BWT: backward transfer,

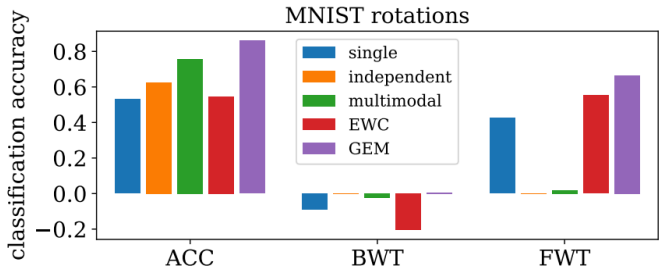
FWT: forward transfer

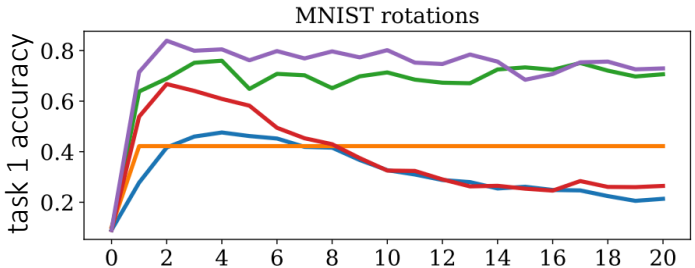
Total memory size:

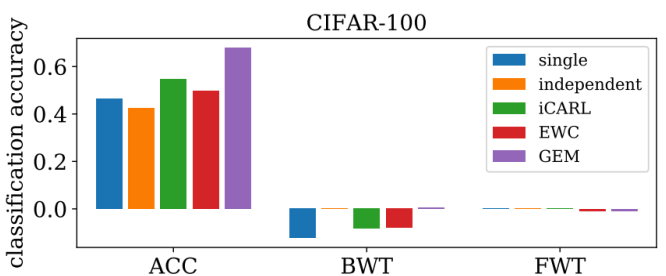
5012 examples

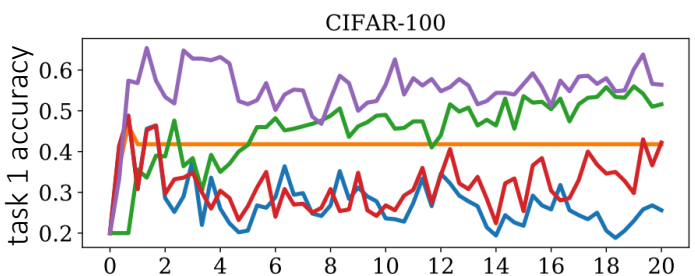












In practice we can't store the data in RAM/VRAM, so the above results may not apply. → 기대효과

If we take a step back… do these experimental domains make sense?

Lopez-Paz & Ranzato. Gradient Episodic Memory for Continual Learning. NeurIPS '17

Can we meta-learn how to avoid negative backward transfer?

Update rules s.t. it doesn't get ↪ve backwards transfer

Javed & White. ***Meta-Learning Representations for Continual Learning.*** NeurIPS ‘19

Beaulieu et al. ***Learning to Continually Learn.*** ‘20

Plan for Today

The lifelong learning **problem statement**

Basic approaches to lifelong learning

Can we do **better** than the basics?

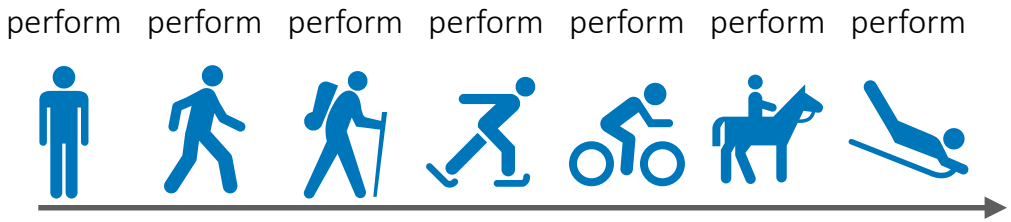
Revisiting the problem statement
from **the meta-learning perspective**

Formulation of online learning when faced with sequence of tasks

Online Learning

(Hannan '57, Zinkevich '03)

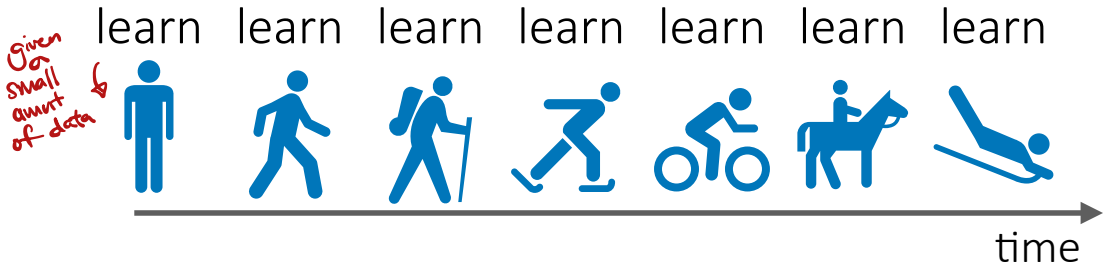
Perform sequence of tasks while minimizing static regret.



zero-shot performance

↳ often difficult to perform well zero-shot on a completely new task.

More realistically:



slow learning → rapid learning

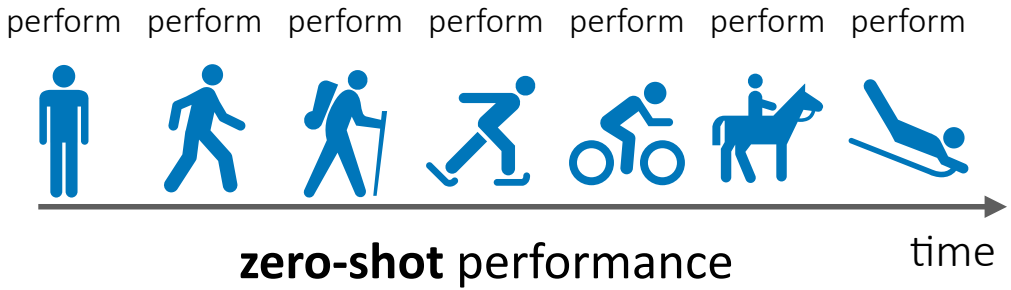
Faster learning over time

Formulation of online learning when faced with sequence of tasks

Online Learning

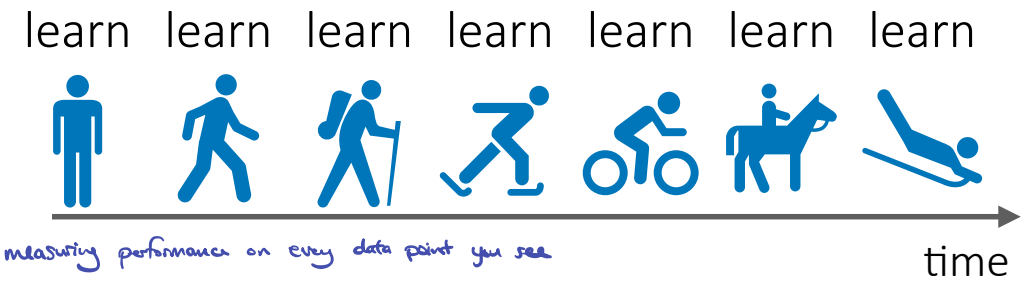
(Hannan '57, Zinkevich '03)

Perform sequence of tasks
while minimizing static regret.



Online Meta-Learning

Efficiently learn a sequence of tasks
from a non-stationary distribution.



evaluate performance after seeing a small amount of data

Primarily a difference in ***evaluation***, rather than the ***data stream***.

(Finn*, Rajeswaran*, Kakade, Levine ICML '18)

The Online Meta-Learning Setting

for task t = 1, ..., n

observe $\mathcal{D}_t^{\text{tr}}$

use update procedure $\Phi(\theta_t, \mathcal{D}_t^{\text{tr}})$ to produce parameters ϕ_t

Initially given a period to try to learn the task w/ a small amount of data

observe x_t

predict $\hat{y}_t = f_{\phi_t}(x_t)$

observe label y_t

Standard online learning setting → Identical

Goal: Learning algorithm with sub-linear

Loss of algorithm

Loss of best algorithm in hindsight

$$\text{Regret}_T := \sum_{t=1}^{T} \ell_t(\Phi_t(\theta_t)) - \min_{\theta \in \Theta} \sum_{t=1}^{T} \ell_t(\Phi_t(\theta))$$

(Finn*, Rajeswaran*, Kakade, Levine ICML '18)

Can we apply meta-learning in lifelong learning settings?

Recall the **follow the leader** (FTL) algorithm:

Store all the data you've seen so far, and train on it.

Deploy model on current task.

Follow the *meta*-leader (FTML) algorithm:

Warm-start the metaparams w/ the metaparams from the previous time step.

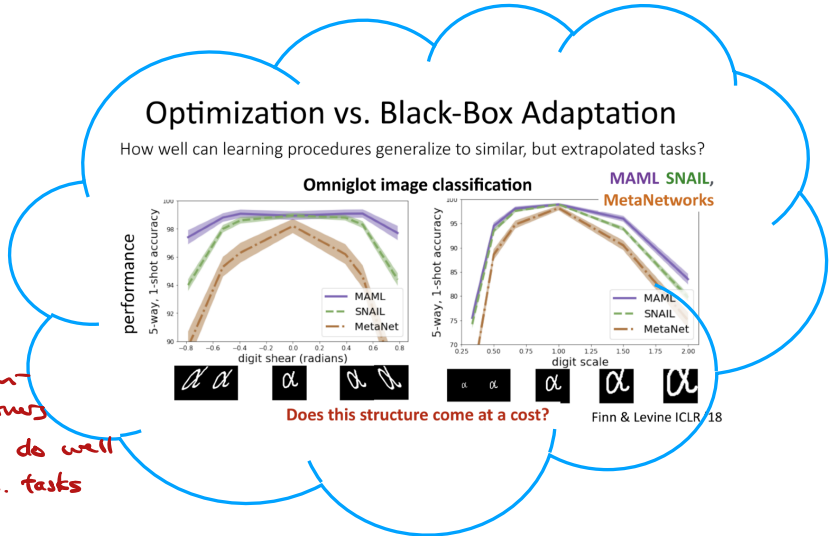
Store all the data you've seen so far, and **meta**-train on it.

Run update procedure on the current task.

What meta-learning algorithms are well-suited for FTML?

What if $p_t(\mathcal{T})$ is non-stationary?

Use optimization-based meta learning b/c it's expected to do well on out-of-distrib. tasks

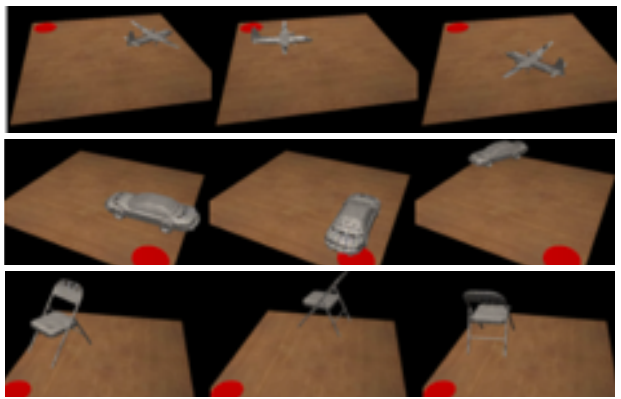


Online meta-learning experiments

Experiment with **sequences of tasks**:

- Colored, rotated, scaled **MNIST**
- **3D object pose prediction**
- **CIFAR-100** classification

Example pose prediction tasks



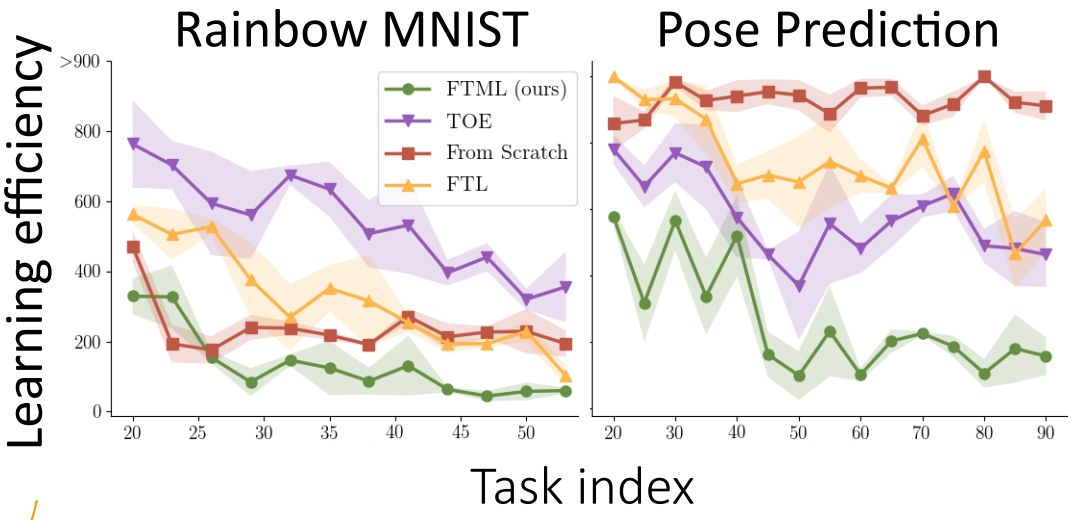
plane

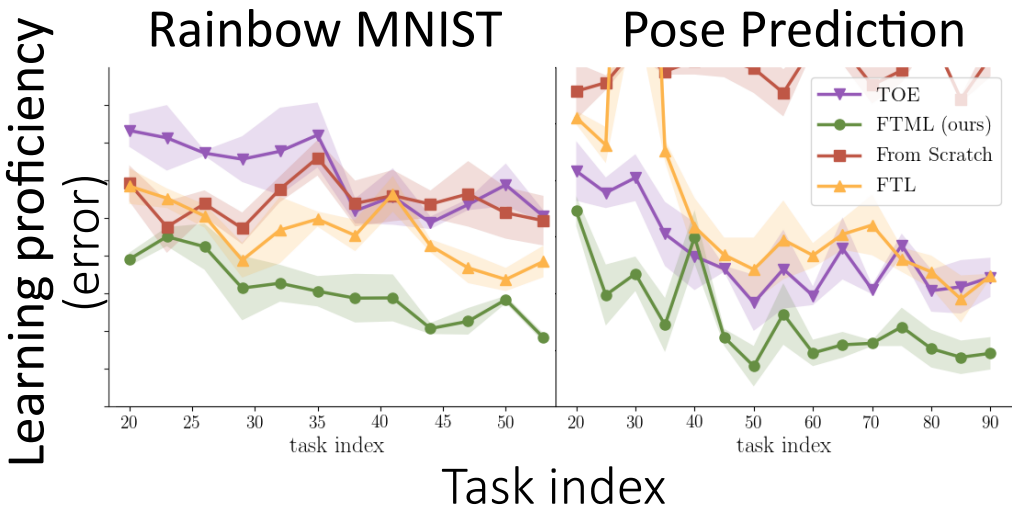
car

chair

Online meta-learning experiments

Comparisons:
- **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





Meta-learning approaches applicable to online learning

Follow The Meta-Leader

learns each new task faster & **with greater proficiency**,

approaches **few-shot learning** regime

Takeaways

Many flavors of lifelong learning, all under the same name.

Defining the problem statement is often the hardest part

Meta-learning can be viewed as a slice of the lifelong learning problem.

A very open area of research.

Course Reminders

Optional homework 4 due **today**.

Project milestone due **Wednesday.**

Guest lecture on Wednesday!

Hanie Sedghi

Please try to show up in person & on-time.