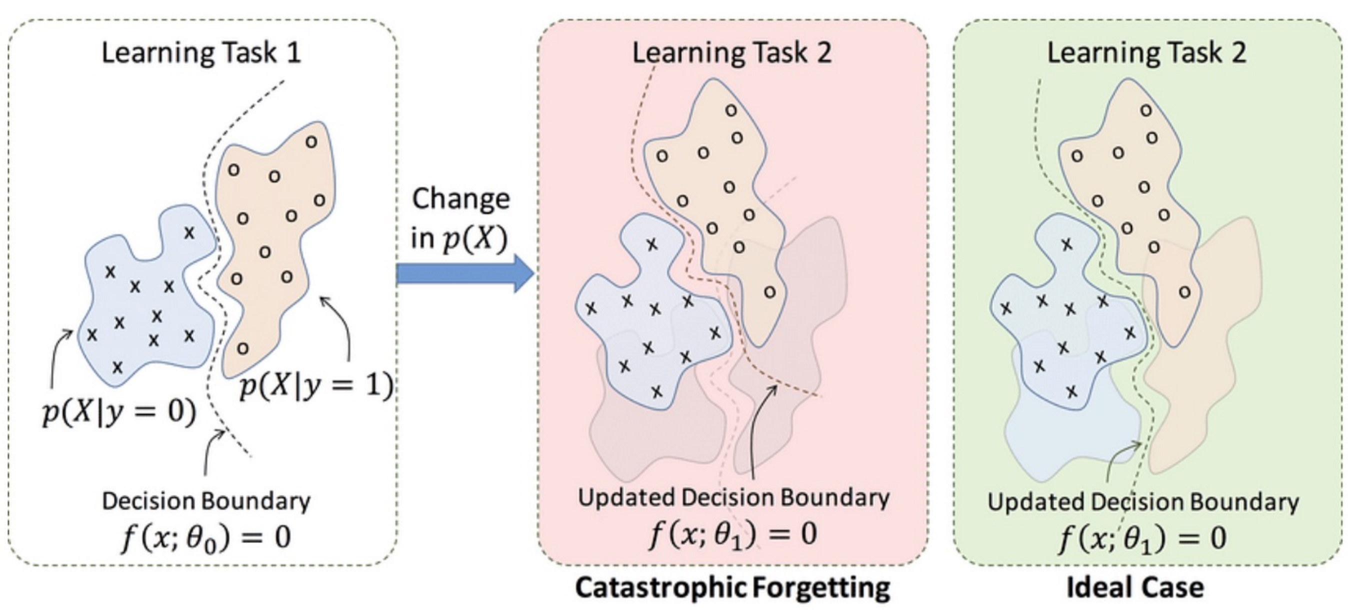
Lifelong Learning for Mobile Robot Task Completion (LLfTC)

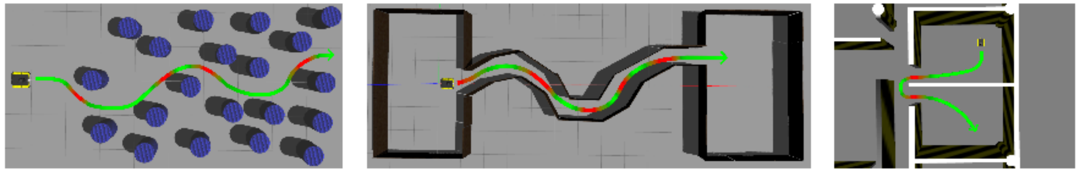
Scott Sikorski
University of Virginia

Motivation

Main Goal: Prevent Catastrophic Forgetting

Second Goal: Take previous work in lifelong learning and extend it to the task domain





(a) Env. 1 (b) Env. 2 (c) Env. 3

Fig. 3: Simulated Navigation Environments: Green segments are primarily traversed using the initial policy π_0, while red segments are mostly traversed using the learned planner π_θ.

Background – Gradient Episodic Memory

We aim to optimize:

$$min_\theta \; l(\pi_\theta, \varepsilon_k) \; s.t. \; l(\pi_\theta, M) \leq l(\pi_{\theta_{k-1}}, M), \forall M \in B$$

Where *k* is the current environment index, *M* is a few example data points for an environment, and *B* is the collection of the k-1 *M's*

$$Define \; l(\pi, x) = \mathbb{E}_{(s,a)\in x} ||\pi_\theta(s) - a||_2$$

And the constraints are satisfied iff θ is initialized from θ_{k-1} and l doesn't increase

So now we optimize:

$$min_\theta \; l(\pi_\theta, \varepsilon_k) \; s.t. < \frac{\partial l(\pi_\theta, \varepsilon_k)}{\partial \theta}, \frac{\partial l(\pi_\theta, M)}{\partial \theta} > \geq 0, \forall M \in B$$

And can use a quadratic program solver without dynamically expanding parameter space

Training

1. Init environment and goal task
2. Pick action using RRT
3. Execute action through controller
4. If successful, add (state, action) to Buffer stream set
5. Check if action completed goal task
6. Set new state
7. Repeat 2 – 6 until completion or timeout
8. Advance environment and go to 1

RRT for Tasks

State

```
self.agentX = envMD['agent']['position']['x’]
self.agentY = envMD['agent']['position']['y']
self.agentZ = envMD['agent']['position']['z']
self.visibleObjects =
        [obj for obj in envMD['objects'] if obj['visible']]
self.visObjName =
        [obj['objectId'] for obj in self.visibleObjects]
self.reachableObjects =
        self.mapPosToObjs(reachMD)
self.reachObjName =
        [obj['objectId'] for obj in self.reachableObjects]
```

Goal Tasks

```
{
"objectId": "Apple|+01.83|+00.78|-00.65",
  "status": "PickUp” },
{
"objectId": "Apple|-01.65|+00.81|+00.07",
  "status": "Move",
  "objPosition": { "x": -1, "y": 0.81, "z": 0.5 }
}
```

- Heuristically interact if a goal object is reachable
- Else, randomly explore
  - Movement in one dimension
  - Rotating counts as an action
- If there are reachable objects, robot can also choose to randomly interact with it
  - Training helps prevent undoable actions like breaking
  - Does not alleviate this issue though

Updating Θ

$$\forall s_k,\ s_k \in M_{i,k}\ s.t\ k \in [0, 200)$$

Task *t*

$$\forall a_k,\ a_k \in M_{i,k}\ s.t\ k \in [0, 200)$$

GEM

- Find memory offset in memory data
- Compute Gradient on 0 … *t* – 1 tasks
- Gather CE Loss on previous examples
- Ensure that CE Loss does not increase with new trained Θ

Θ

Model Design

- GEM: FFC MLP using Cross Entropy Loss and SGD optimizer
  - 2 layers, 100 hidden nodes, 0.001 learning rate
- Training Data: (State, Action) pairs for current environment
- Epochs: 10
- Timeout: 100s
- Number of Memories: 300

Testing

1. Init environment and goal
2. Get action from RRT and model
3. Execute action with highest benefit
4. If successful, add (state, action) to transition stream set
5. Check if action completed goal task
6. Set new state
7. Repeat 2 – 6 until completion or timeout
8. Advance environment and go to 1

Scoring States

```
Diff(s1, s2):
    d += √Σ(s_{1,i} − s_{2,i})²
    inBoth = s1.obj ∩ s2.obj
    d += 10 * (|s1.obj|+|s2.obj|-inBoth)
    for obj in inBoth:
        d += √Σ(obj_{1,i} − obj_{2,i})²
```

$$d \mathrel{+}= \sqrt{\sum (s_{1,i} - s_{2,i})^2}$$

$$d \mathrel{+}= \sqrt{\sum (obj_{1,i} - obj_{2,i})^2}$$

Results

Environment	# of LLfTC actions	# of RRT actions	Completed
1	4	10	T
2	200	200	F
3	50	134	T
4	51	99	T
5	200	200	T

Table 1: Testing Results to Pickup Apple

Avg Number of Completions	Avg # of fails	Avg Training Actions
1	1	8
0	183	200
0.1	22	155
0.8	5	120
0	200	200

Table 2: Training Results to Pickup Apple

Limitations & Improvements

- Incorporating RL into the model
  - Making GEM compatible with a Q function
- Feeding GEM a better defined model
  - 2017 model, we can do a lot better
- If training does not find the correct action to complete a task, we won't finish
  - Adapt with state to find a better direction even if state isn't completely the same
  - Use/develop a better sampling policy that integrates well with task completion

Q & A

Citations

Liu, Bo, Xuesu Xiao, and Peter Stone. "A lifelong learning approach to mobile robot navigation." *IEEE Robotics and Automation Letters* 6.2 (2021): 1090-1096.

Lopez-Paz, David, and Marc'Aurelio Ranzato. "Gradient episodic memory for continual learning." *Advances in neural information processing systems* 30 (2017).

https://github.com/facebookresearch/GradientEpisodicMemory/tree/master

https://github.com/sgsikorski/LLfTC