One-Shot Visual Imitation Learning via Meta-Learning

Chelsea Finn\textsuperscript{1}, Tianhe Yu\textsuperscript{1}, Tianhao Zhang\textsuperscript{1}, Pieter Abbeel\textsuperscript{1,2}, Sergey Levine\textsuperscript{1}

2017

\textsuperscript{1} UC Berkley and \textsuperscript{2} OpenAI
Builds on

Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks

Chelsea Finn¹, Pieter Abbeel¹,², Sergey Levine¹

2017

¹ UC Berkley and ² OpenAI
One-Shot Visual Imitation Learning via Meta-Learning

- What is imitation learning?
One-Shot Visual Imitation Learning via Meta-Learning

What is One-Shot learning?

- Learning to generalize from a single example $(x, y)$
  - $X$ - observation (state, image, etc)
  - $Y$ - output (action, class-label, etc)

What is Meta-Learning?

- “Learning to learn” - training a model on a variety of tasks such that it can solve new tasks using a small number of samples [1].
How do we solve new tasks with few samples?

Leverage information from previous tasks to more quickly solve new tasks

Traditional approach (single task)

Meta-approach (multiple tasks)
Why bother?

Humans Can Learn New Tasks Quickly!

For example, humans can learn to identify “novel two-wheel vehicles” from a single picture (e.g. as shown on the right), whereas machines cannot generalize a concept from just a single image [3].

And We Want Machines To Learn Quickly Too
Primary paper contribution

- Demonstrates that Model-Agnostic Meta-Learning (MAML) can be used to train visual motor policies that can adapt to new tasks with only one visual demonstration [2].
Pick-and-Place Results

Demo
Task 2
real time

Contextual
LSTM
DAML, linear loss
DAML, temporal loss (ours)
How does MAML work?

- Learns initializations that can easily be fine-tuned for new tasks

\[
\theta_i' = \theta - \alpha \nabla_{\theta} \mathcal{L}_{T_i}(f_{\theta})
\]

Images taken from [2]
Meta-learning objective

\[
\min_{\theta} \sum_{T_i \sim p(T)} \mathcal{L}_{T_i}(f_{\theta}') = \sum_{T_i \sim p(T)} \mathcal{L}_{T_i}(f_{\theta} - \alpha \nabla_{\theta} \mathcal{L}_{T_i}(f_{\theta}))
\]

where

\[
\mathcal{L}_{T_i}(f_\phi) = \sum_{T^{(j)} \sim T_i} \sum_{t} \| f_\phi(o_t^{(j)}) - a_t^{(j)} \|_2^2
\]

Essentially, optimizes the val accuracy of the fine-tuned model w.r.t the initial parameters

Images taken from [2]
Meta-training loop

**Algorithm 1** Meta-Imitation Learning with MAML

**Require:** $p(T)$: distribution over tasks  
  
**Require:** $\alpha$, $\beta$: step size hyperparameters

1: randomly initialize $\theta$
2: while not done do
3:   Sample batch of tasks $T_i \sim p(T)$
4:   for all $T_i$ do
5:     Sample demonstration $\tau = \{o_1, a_1, \ldots, o_T, a_T\}$ from $T_i$
6:     Evaluate $\nabla_\theta L_{T_i}(f_\theta)$ using $\tau$ and $L_{T_i}$ in Equation (2)
7:     Compute adapted parameters with gradient descent: $\theta'_i = \theta - \alpha \nabla_\theta L_{T_i}(f_\theta)$
8:     Sample demonstration $\tau'_i = \{o'_1, a'_1, \ldots, o'_T, a'_T\}$ from $T_i$ for the meta-update
9:   end for
10:  Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{T_i \sim p(T)} L_{T_i}(f_{\theta'_i})$ using each $\tau'_i$ and $L_{T_i}$ in Equation 2
11: end while
12: return parameters $\theta$ that can be quickly adapted to new tasks through imitation.

*Images taken from [2]*
Experiments

1. Simulated Reaching
2. Simulated Pushing
3. Real-World Placing

Images taken from [2]
Learning to imitate without expert actions (video only)

\[ L_{T_i}(f_{\phi}) = \sum_{\tau(j) \sim T_i} \sum_t \| f_{\phi}(o_t^{(j)}) - a_t^{(j)} \|_2^2 \]

\[ L_{T_i}^*(f_{\phi}) = \sum_{\tau(j) \sim T_i} \sum_t \| W y_t^{(j)} + b \|_2^2 \]

Images taken from [2]
Baselines

- **Random policy**: A policy that outputs random actions from a standard Normal distribution.
- **Contextual policy**: A feedforward policy, which takes as input the final image of the demonstration, to indicate the goal of the task, and the current image, and outputs the current action.
- **LSTM**: A recurrent neural network which ingests the provided demonstration and the current observation, and outputs the current action, as proposed by Duan et al.
- **LSTM+attention**: A recurrent neural network using the attention architecture proposed by Duan et al. (Note: this method only applies to non-vision tasks)

text taken from [2]
Simulated reaching

Images taken from [2]
Simulated pushing

<table>
<thead>
<tr>
<th>method</th>
<th>video+state +action</th>
<th>video +state</th>
<th>video</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSTM</td>
<td>78.38%</td>
<td>37.61%</td>
<td>34.23%</td>
</tr>
<tr>
<td>contextual</td>
<td>n/a</td>
<td>58.11%</td>
<td>56.98%</td>
</tr>
<tr>
<td>MIL (ours)</td>
<td><strong>85.81%</strong></td>
<td><strong>72.52%</strong></td>
<td><strong>66.44%</strong></td>
</tr>
<tr>
<td>LSTM</td>
<td>83.11%</td>
<td>39.64%</td>
<td>31.98%</td>
</tr>
<tr>
<td>contextual</td>
<td>n/a</td>
<td>64.64%</td>
<td>59.01%</td>
</tr>
<tr>
<td>MIL (ours)</td>
<td><strong>88.75%</strong></td>
<td><strong>78.15%</strong></td>
<td><strong>70.50%</strong></td>
</tr>
</tbody>
</table>

Total number of demonstrations in the meta-training set
Real-world placing

<table>
<thead>
<tr>
<th>method</th>
<th>test performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSTM</td>
<td>25%</td>
</tr>
<tr>
<td>contextual</td>
<td>25%</td>
</tr>
<tr>
<td>MIL</td>
<td>90%</td>
</tr>
<tr>
<td>MIL, video only</td>
<td>68.33%</td>
</tr>
</tbody>
</table>

Table 2: One-shot success rate of placing a held item into the correct container, with a real PR2 robot, using 29 held-out test objects. Meta-training used a dataset with ~100 objects. MIL, using video only receives the only video part of the demonstration and not the arm trajectory or actions.

Images and video taken from [2]
Questions?
Sources

