Self-Supervision and Play

Pierre Sermanet

In collaboration with
Corey Lynch, Debidatta Dwibedi, Soeren Pirk, Jonathan Tompson, Mohi Khansari, Yusuf Aytar, Yevgen Chebotar, Yunfei Bai, Jasmine Hsu, Eric Jang, Vikash Kumar, Ted Xiao, Stefan Schaal, Andrew Zisserman, Sergey Levine

Robotics at Google
http://g.co/robotics
Main Message

- Real-world robotics cannot rely on labels and rewards
- Instead, mostly
  - Self-supervise on unlabeled data
  - Use play data
- We present ways to do this for vision and control
Our mission:

Self-Supervised Robots

i.e.: Autonomously extract learning signals from the world from play and from others

because: “Give a robot a label and you feed it for a second; Teach a robot to label and you feed it for a lifetime.”
Supervision Types

**Supervised**
- Signal: Discrete & human-defined
- Embedding
  - 1. Hand pose
  - 2. Cups pose
  - 3. Liquid amounts
  - 4. Liquid flowing
  - 5. Liquid color

**Unsupervised**
- Signal: Continuous embedding
  - Auto-encoding
  - Sparsity prior
  - Gaussian prior

**Self-Supervised**
- Signal: Continuous embedding
  - Depth
  - Time
  - Multi-view
  - Tactile feedback
## Supervision Costs

<table>
<thead>
<tr>
<th>Type of Supervision</th>
<th>Description</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Playing (Intrinsic Motivation)</td>
<td>Alone or with others</td>
<td>Free</td>
</tr>
<tr>
<td>Play data (Tele-op)</td>
<td></td>
<td>Very cheap</td>
</tr>
<tr>
<td>Imitation</td>
<td>other agents “playing” for hours, not segmented, not labeled</td>
<td>Cheap (but not unlimited)</td>
</tr>
<tr>
<td>Demonstrations</td>
<td>Staged, segmented and labeled</td>
<td>Expensive</td>
</tr>
<tr>
<td>Labeled Frames</td>
<td>e.g. action and object classes / attributes</td>
<td>Very Expensive</td>
</tr>
</tbody>
</table>
Why Self-Supervise?

- Be versatile and robust to **different hardware & environments:**
  - Robot-agnostic and self-calibrating
  - Agnostic to sim or real, train the same way
- **Scaling up** in the real world
- Can’t afford human supervision given the high dimensionality of the problem
  - Labeling is not easy to define even for humans
- **Rich representations** can be discovered through self-supervision and lead to **higher sample efficiency** in RL
Why play?

- Self-Supervision enables using play data
- Cheap
- General
- Rich
label free
Self-Supervision and Play for Vision
Self-Supervised Visual Representations

- Time-Contrastive Networks (TCN)
- Temporal Cycle-Consistency (TCC)
- Object-Contrastive Networks (OCN)

\[ \text{disentangled / invariant states and attributes} \]

Imitation

Progression of task
Imitation

Progression of task

Play

- Detect errors
- Correct
- Retry
- Transition
- General skills
- Data augmentation
Time-Contrastive Networks (TCN)

[ Sermanet*, Lynch*, Chebotar*, Hsu, Jang, Schaal, Levine @ ICRA 2018 ] [ sermanet.github.io/imitate ]
Single-view TCN
Semantic Alignment with TCN

Observation  multi-view TCN
Robotic Imitation:
Step 1. Self-Supervise on Play data
Robotic Imitation:
Step 2. Follow abstract trajectory
Actionable Representations

[ Dwibedi, Tompson, Lynch, Sermanet @ IROS 2018 ] [ sites.google.com/view/actionablerepresentations ]
Cheetah Environment

Agent observes another agent demonstrating an action
Qualitative Results: Cheetah

<table>
<thead>
<tr>
<th>Input to PPO</th>
<th>Cumulative Reward (Avg of 100 runs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random State</td>
<td>28.31</td>
</tr>
<tr>
<td>True State</td>
<td>390.16</td>
</tr>
<tr>
<td>Raw Pixels</td>
<td>146.14</td>
</tr>
<tr>
<td>mfTCN</td>
<td>360.50</td>
</tr>
</tbody>
</table>

- PPO on true state
- PPO on learned visual representations
Temporal Cycle-Consistency (TCC)

[ Dwibedi, Aytar, Tompson, Sermanet, Zisserman @ CVPR 2019 ] [ temporal-cycle-consistency.github.io ]
Temporal Cycle-Consistency (TCC)

[ Dwibedi, Aytar, Tompson, Sermanet, Zisserman @ CVPR 2019 ] [ temporal-cycle-consistency.github.io ]
Temporal Cycle-Consistency

- Cycle consistency error
- Nearest neighbors
- Cycle consistent
- Not cycle consistent

embedding space
embedding space
reduced to 2D via t-SNE

reference video

moving down

moving up
Action Phase Classification

<table>
<thead>
<tr>
<th>Datasets</th>
<th>% of Labels</th>
<th>0.1</th>
<th>0.5</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Supervised Learning</td>
<td>57.69</td>
<td>78.55</td>
<td>83.83</td>
</tr>
<tr>
<td></td>
<td>SaL [18]</td>
<td>70.78</td>
<td>74.39</td>
<td>76.35</td>
</tr>
<tr>
<td></td>
<td>TCN [25]</td>
<td>80.21</td>
<td>81.77</td>
<td>82.52</td>
</tr>
<tr>
<td></td>
<td>TCC (ours)</td>
<td>76.41</td>
<td>79.85</td>
<td>81.68</td>
</tr>
<tr>
<td></td>
<td>TCC + SaL (ours)</td>
<td>77.90</td>
<td>81.39</td>
<td>83.11</td>
</tr>
<tr>
<td></td>
<td>TCC + TCN (ours)</td>
<td><strong>81.59</strong></td>
<td><strong>83.50</strong></td>
<td><strong>84.11</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Datasets</th>
<th>% of Labels</th>
<th>0.1</th>
<th>0.5</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Supervised Learning</td>
<td>77.31</td>
<td>85.42</td>
<td>90.12</td>
</tr>
<tr>
<td></td>
<td>SaL [18]</td>
<td>79.36</td>
<td>86.62</td>
<td>86.72</td>
</tr>
<tr>
<td></td>
<td>TCN [25]</td>
<td>86.94</td>
<td>88.51</td>
<td>89.14</td>
</tr>
<tr>
<td></td>
<td>TCC (ours)</td>
<td>84.73</td>
<td>88.83</td>
<td><strong>91.45</strong></td>
</tr>
<tr>
<td></td>
<td>TCC + SaL (ours)</td>
<td>87.80</td>
<td>89.55</td>
<td>90.61</td>
</tr>
<tr>
<td></td>
<td>TCC + TCN (ours)</td>
<td><strong>90.97</strong></td>
<td><strong>90.17</strong></td>
<td>90.33</td>
</tr>
</tbody>
</table>

Phase classification results when fine-tuning ImageNet pre-trained ResNet-50.
Self-Supervised Alignment
Object-Contrastive Networks (OCN)

[ Pirk, Khansari, Bai, Lynch, Sermanet @ under review ]

Self-teaching about any object leads to high robustness, allowing deployment
Robotic Data Collection
Play data for Objects
Object-Contrastive Networks (OCN)

[ Pirk, Khansari, Bai, Lynch, Sermanet @ under review ]
Recovering Continuous Attributes

(Same instance removed)
Online Object Understanding

- Offline average error: 54%
- Online average error: 17% -> 3%
- Do not define states and attributes
Online Adaptation

ResNet50 - 50.6% error

Training on the first 5 seconds...
Online Adaptation

ResNet50 - 81.9% error

Trained on 160s - 40.3% error
Self-Supervision and Play for Control
Pose Imitation with TCN

[ Sermanet*, Lynch*, Chebotar*, Hsu, Jang, Schaal, Levine @ ICRA 2018 ] [ sermanet.github.io/imitate ]
Play Data in Pose Space

Human Play

Robot Scripting

Human imitating Robot
Pose Imitation with Play Data

Model

- TCN supervision
- TCN supervision
- Self-regression

Human Play

Robot Script

Robot Script

internal joints
Pose Imitation

- Self-Supervision + Play recipe
- No explicit task definition.
Learning from Play (LfP)

[ Lynch, Khansari, Xiao, Kumar, Tompson, Levine, Sermanet @ under review ] [ learning-from-play.github.io ]

- No tasks
- No rewards or RL
- Multiple tasks in zero-shot
- 85% on 18 tasks
- Self-Supervision + Play recipe
Continuum of skills
How can we cover the continuum?

Scripted collection + RL

Exploration

Scripting exploration

Reward sensor for door opening

Reward engineering

Reset

Distributed training
Tasks are not discrete

“Grasp fast?”

“Nudge slow?”

“Nudge + grasp?”

Slide “full”? 

Slide “partial”? 

Boundaries between multiple tasks?
How can we cover the continuum?

Learning from Demonstration (LfD)

Kinesthetic
[Kober and Peters, 2011]

Tele-op, segmented expert demonstrations
How can we cover the continuum?

Learning from Play (LfP)

Learning from Demonstration (LfD)

Scripted collection + RL
Play data for training
collected from human tele-operation

(2.5x speedup)
How do we learn control from play?

- **current**
- **goal-conditioned policy**
- **goal**
Play covers the continuum
Goal relabeling

600 unique sequences, 3.65 hours

60 seconds
Multimodality issue
1. Given **unlabeled play data**

2. Learn **latent plans** using self-supervision

3. Decode plan to reconstruct actions

**Training Play-LMP**
Play-LMP: Test time

1. Given a goal state

2. Generate a latent plan (@1Hz)

3. Generate an action (@30Hz)
18 tasks (for evaluation only)

close drawer close sliding open drawer grasp flat grasp lift grasp upright
knock pull out shelf push blue push green push red put in shelf
rotate left rotate right sliding sweep sweep left sweep right
Quantitative Accuracy

We obtain a single task-agnostic policy and evaluate it on 18 zero-shot tasks.

- **Play-LMP**: single policy trained on cheap unlabelled data: 85% zero shot
- **Baseline**: 18 policies trained on expensive labelled data: 65%
- When perturbing the start position, the success is:
  - baseline: 23%
  - Play-LMP: 79%
Examples of success runs for Play-LMP

Goal
(task: sliding)

Play-LMP policy
Examples of success runs for Play-LMP

Goal
(task: sweep)

Play-LMP policy
Examples of success runs for **Play-LMP**

Goal
(task: pull out of shelf)

Play-LMP policy
Examples of success runs for **Play-LMP**

Goal
(task: rotate left)

Play-LMP policy
Some failure cases for Play-LMP

Goal
(task: sliding)

Play-LMP policy
Some failure cases for Play-LMP

Goal
(task: pull out of shelf)

Play-LMP policy
Retrying behavior emerging from Play-LMP
Retrying behavior emerging from Play-LMP

Goal
(task: pull out of shelf)

Play-LMP policy
Retrying behavior emerging from Play-LMP

Goal
(task: sweep right)

Play-LMP policy
Composing 2 skills: grasp + close drawer

Goals

Play-LMP policy
Composing 2 skills: put in shelf + close sliding

Goals

Play-LMP policy
Composing 2 skills: open sliding + push green

Goals

Play-LMP policy
Composing 2 skills: sweep + close drawer

Goals

Play-LMP policy
Composing 2 skills: drawer open + sweep

Goals

Play-LMP policy
8 skills in a row

Goal

Play-LMP policy
Latent plan space (t-SNE)
Richness & Scalability of Data

- Learning from demonstrations (LfD)
- Learning from play (LfP)
- Scripted collection + RL

Richness

Scalable
Recipe: **Self-Supervision + Play**
Takeaways

- **Self-Supervision + Play** recipe:
  - Self-supervise on lots of unlabeled data
  - Use play data

- **Delay definitions** of tasks, states or attributes,
  Let self-supervision organize continuous spaces:
  - Continuum of states and attributes
  - Continuum of skills
Questions?

g.co/robotics
sermanet.github.io
sermanet@google.com