Sim-To-Real Robot Learning with Progressive Nets

Raia Hadsell

Google DeepMind
Progressive Neural Networks
Progressive Neural Networks
Progressive Neural Networks

\[ \pi_1 \rightarrow \nu_1 \rightarrow \pi_2 \rightarrow \nu_2 \rightarrow \pi_3 \rightarrow \nu_3 \]
Progressive Neural Networks

Advantages

1. No catastrophic forgetting of previous tasks - by design.
2. Deep, compositional feature transfer from all previous tasks and layers
3. Added capacity for learning task-specific features
4. Provides framework for analysis of transferred features

Disadvantages

1. Requires knowledge of task boundaries
2. Scaling! Overall parameter growth is quadratic in the number of tasks (backward pass grows linearly).
Deep RL for Robotics

- **Deep reinforcement learning** has promise to revolutionise robotics
  - Learning human-level skills directly from raw sensor data
- However, there is a massive data problem
  - State-of-the-art deep RL requires huge amounts of data in the form of interactive environments
- **Progressive nets could be used to transfer learned policies from simulation to robot, even from pixel inputs.**
Simulation vs. Reality

Deep learning and deep RL train very well from simulation:

- Training: simulators run 24/7
- Algorithms: multi-threaded
- Hyperparameters: swept
- Speed: faster than real time

However, simulation is only valuable if whatever is learned can transfer to real robot domain.
Sim-to-Real with Progressive Nets - RAIA HADSELL

Simulation

\[ \pi_1 \rightarrow v_1 \]

Task A

Robot

\[ \pi_2 \rightarrow v_2 \]

Task A

\[ \pi_3 \rightarrow v_3 \]

Task B

Task A

Task B
**Column 1:** Reacher task with random start, random target. Episodes have 50 steps; +1 reward when palm is within 3cm of target.

**Input:** RGB only

**Output:** joint velocities (6 DOF)

**Network:** ConvNet + LSTM + softmax output

**Learning:** Asynchronous advantage actor-critic (A3C); 16 threads
24 hrs on Borg $\Rightarrow \approx 55$ days real robot time
Sim-to-Real with Progressive Nets - RAIA HADSELL

The diagram illustrates the process of sim-to-real adaptation using progressive nets. The notation $\pi_1$, $\pi_2$, and $\pi_3$ represent different stages of the network, with $\nu_1$, $\nu_2$, and $\nu_3$ indicating the corresponding value functions. The numbers 128, 16, and 16 likely refer to the dimensions or layers in the network architecture.

The chart on the right compares the rewards of real-robot-trained progressive nets against baselines. The legend includes curves for wide and narrow columns, both trained progressively and finetuned. The x-axis represents the number of steps, while the y-axis shows the rewards.
Fine-tuning vs. progressive?

**Subtle perspective changes**
- Finetuned
- Progressive

**Significant perspective changes**
- Finetuned
- Progressive

**Subtle color changes**
- Finetuned
- Progressive

**Significant color changes**
- Finetuned
- Progressive
Sim-to-Real with Progressive Nets - RAIA HADSELL

\[ \pi_1 \xrightarrow{\nu_1} \nu_1 \]
\[ \pi_2 \xrightarrow{\nu_2} \nu_2 \]
\[ \pi_3 \xrightarrow{\nu_3} \nu_3 \]

128 proprioception sensor data input to LSTM

Real-robot-trained progressive nets (conveyor task)

\[ \phi \]

\[ \phi \] proprioception sensor data input to LSTM
Matching shades of green is a bit of a pain...

Tried and tested method for improving generalisation:

Data augmentation
Targets are one of 4 geometric shapes with random sizes and randomly placed distractors.

Camera position sampled with gaussian noise in every episode.

Target colors are picked uniformly at random. Random table colors. Random light source height and colors.

www.youtube.com/watch?v=6-Th424dvvk&feature=player_embedded
Robustly trained agents are better in novel environments in simulation...
... and, more importantly, in reality:

![Graph showing average episode scores for baseline and robust agents in reality.](image)
Progressive Neural Networks
Sim-to-Real Robot Learning from Pixels

In collaboration with:

Andrei Rusu  Matej Vecerik  Tom Rotherl  Razvan Pascu  Nicolas Heess

Thank you!