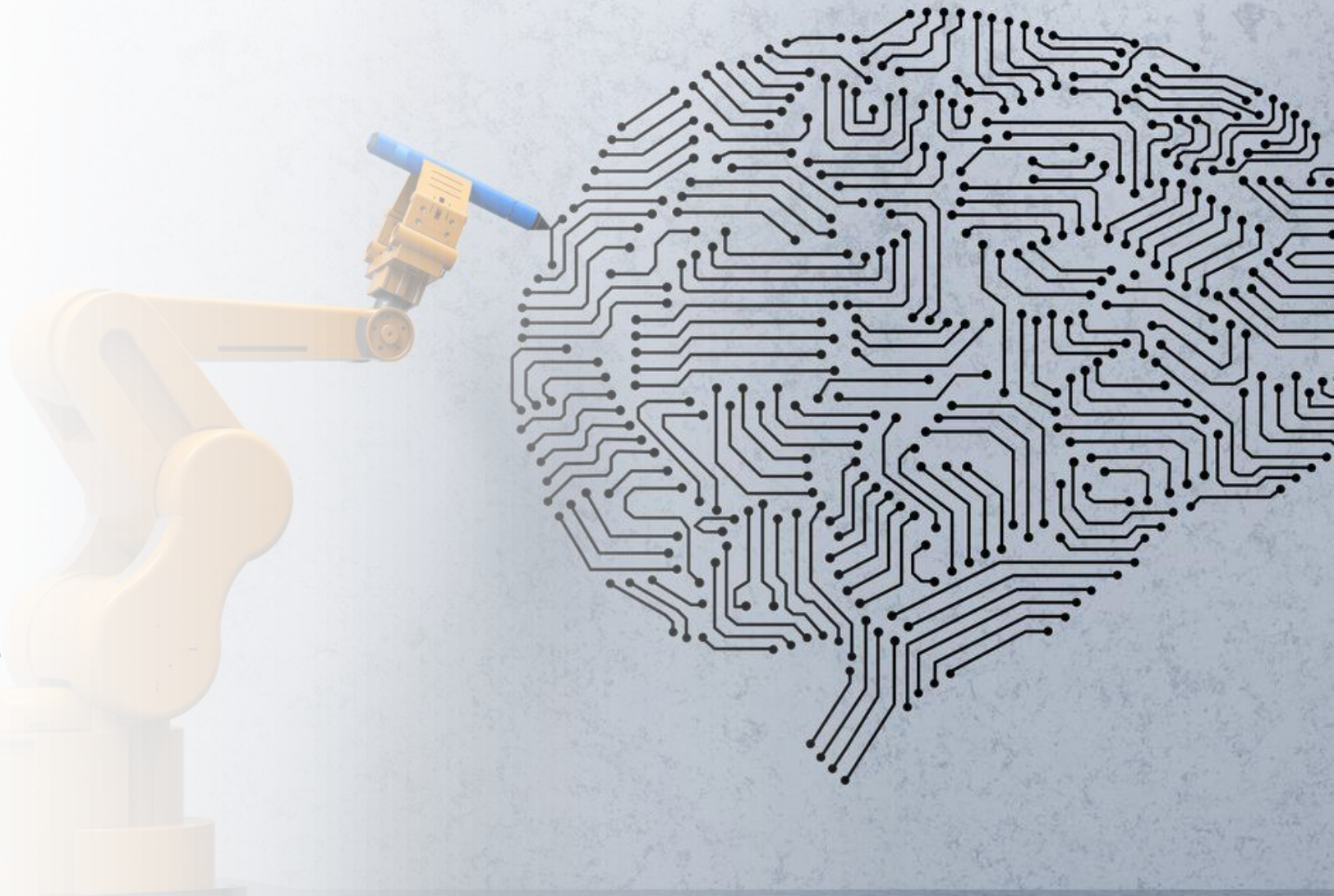
Aprendizaje por refuerzo

Clase 23: RL multi-tarea



Último mes del curso

	3 sesiones	03-17
	Cierre:	17
	Examen 2:	22
	Proyecto:	24

Antes de empezar...

Para el día de hoy…

- Transferir de una tarea a una nueva
- Transferencia hacia adelante
- Multi-tarea
- Política contextual

El problema

- Algunas tareas son fáciles
- Otras son muy difíciles

Venganza de Moctezuma

- Recompensas
  - Obtener la llave
  - Abrir la puerta
- Castigos
  - Morir al tocar la calavera

Venganza de Moctezuma II

- Sabemos que hacer porque entendemos el significado de la imagen
- Sabemos que las llaves abren puertas
- Sabemos usar escaleras
- No sabemos que hace la calavera pero sabemos que no es algo bueno
- El conocimiento previo de la estructura de un problema nos puede ayudar a resolver tareas complejas

¿Puede RL utilizar ese conocimiento previo?

- Si hemos resuelto tareas anteriores, podemos adquirir conocimiento para nuevas tareas
- ¿Cómo podemos almacenar el conocimiento?
  - Funciones Q
  - Políticas
  - Modelos
  - Características

Transferencia de aprendizaje

Utilizar experiencia de un tarea a otra para aprendizaje más rápido o mejor desempeño

En RL una tarea es un MDP

¿Cómo se puede realizar?

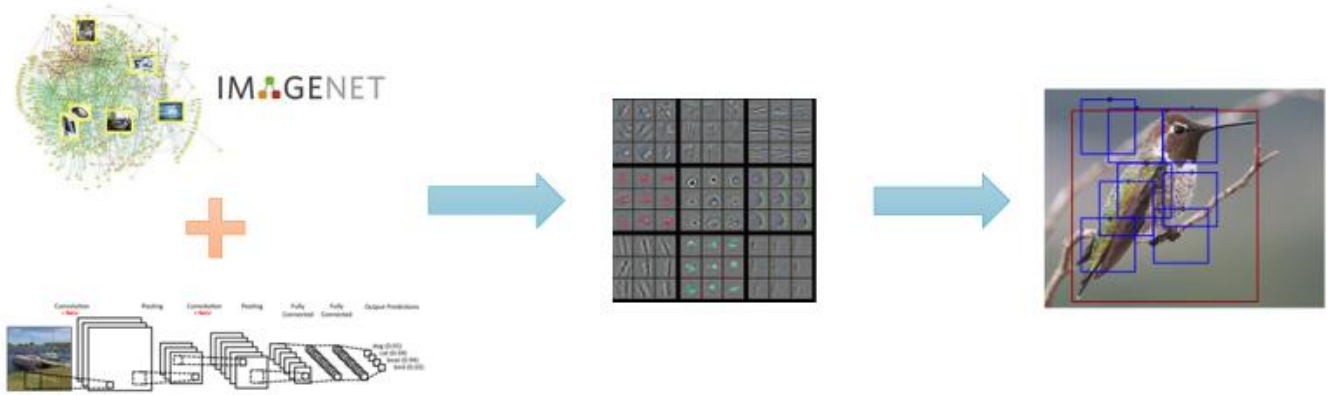
Transferencia hacia adelante	Transferencia multi-tarea	Meta aprendizaje
• Intentarla y esperar lo mejor • Ajuste fino en la nueva tarea • Aleatoriedad en el dominio origen	• Generar dominios origen altamente aleatorios • RL basado en modelo • Destilación de modelo • Políticas contextuales • Redes de política modular	• Basado en gradiente • Basado en redes neuronales recurrentes

Intentarlo y esperar lo mejor

- Las políticas entrenadas para un conjunto de circunstancias pueden trabajar en un nuevo dominio
- No existen garantías

Ajuste fino

- Es el método más popular en aprendizaje profundo supervisado
- Funciona si la tarea origen es muy amplia
- Se entrena la red en la tarea origen
- Se quita la última capa y se agrega una nueva para la tarea objetivo
- Se entrena la última capa o en ocasiones el resto partiendo de los pesos anteriores

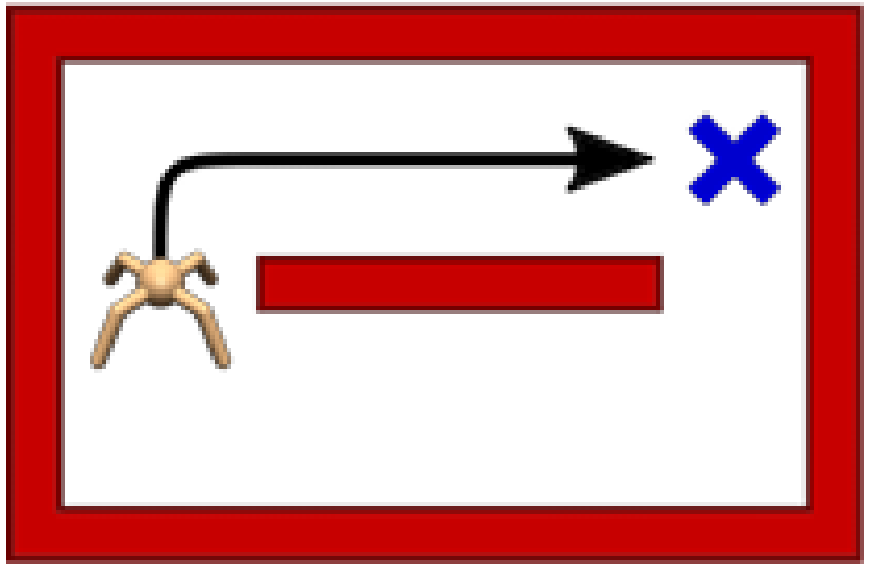


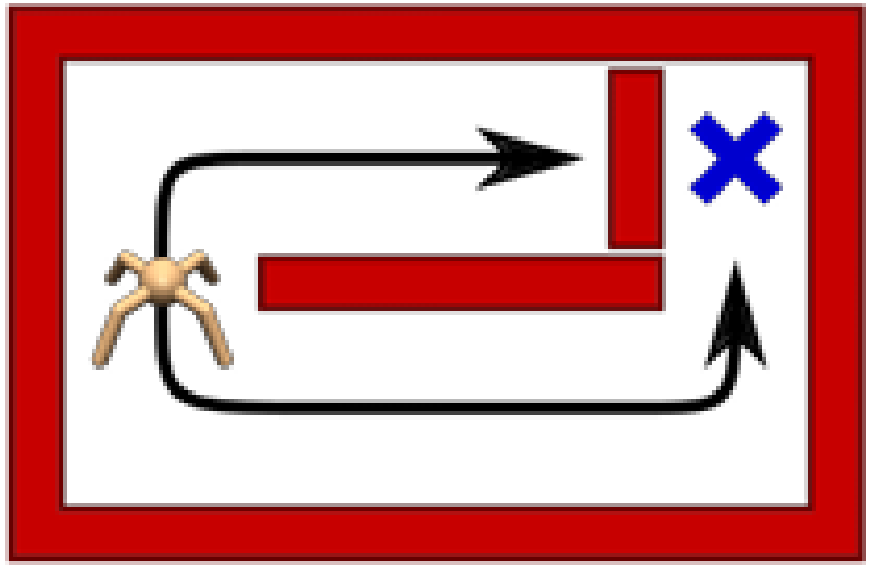
Retos de ajuste fino en RL

- Las tareas de RL son mucho menos diversas
  - Las características son menos generales
  - Las políticas y funciones de valor son muy especializadas
- Las políticas óptimas en MDPs completamente observables son deterministas
  - Se pierde exploración en convergencia
  - Se adapta muy lento en nuevos ambientes

Entrenar para robustez

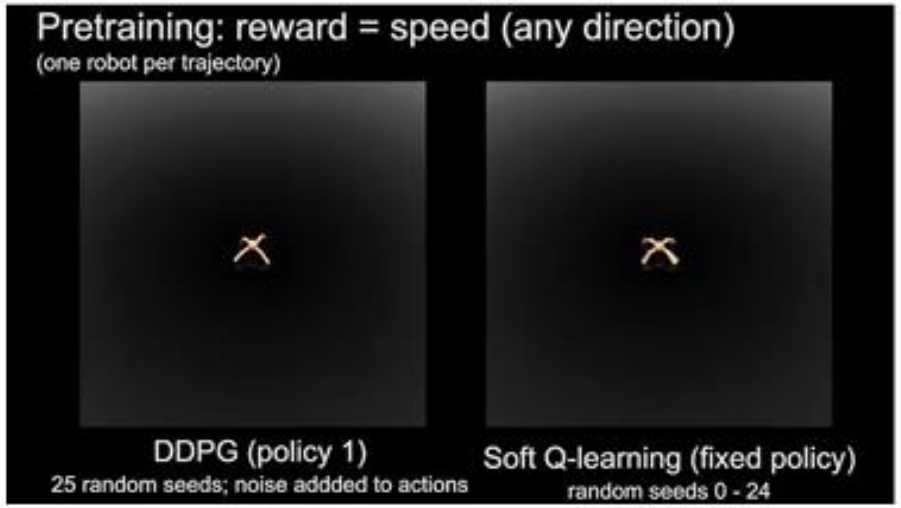
Aprender a resolver una tarea en todas las formas posibles

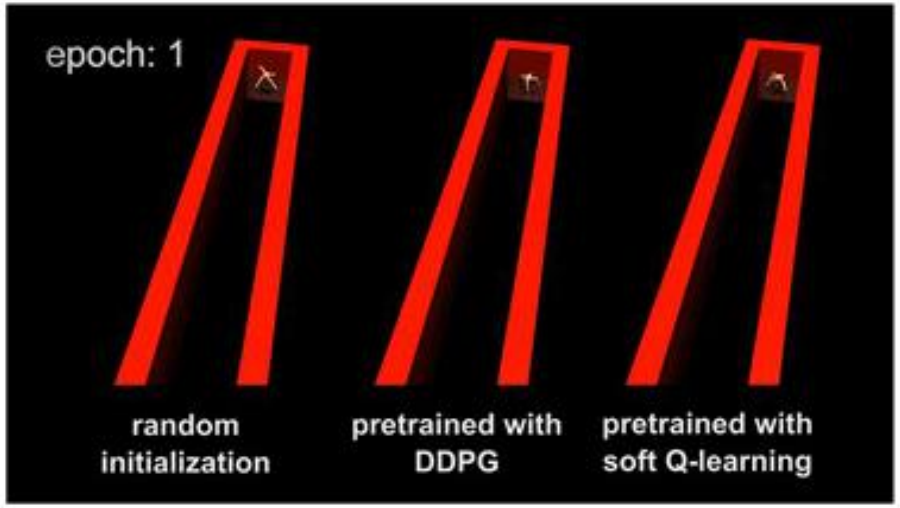


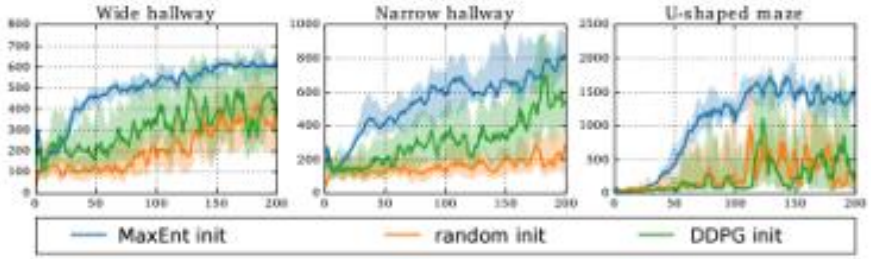


Pre entrenar para diversidad

Haarnoja, Tang et al. Reinforcement learning with Deep energy-based policies







Para saber más

Finetuning via MaxEnt RL: Haarnoja*, Tang*, et al. (2017). **Reinforcement Learning with Deep Energy-Based Policies.**

Finetuning from transferred visual features (via VAE): Higgins et al. **DARLA: improving zero-shot transfer in reinforcement learning.** 2017.

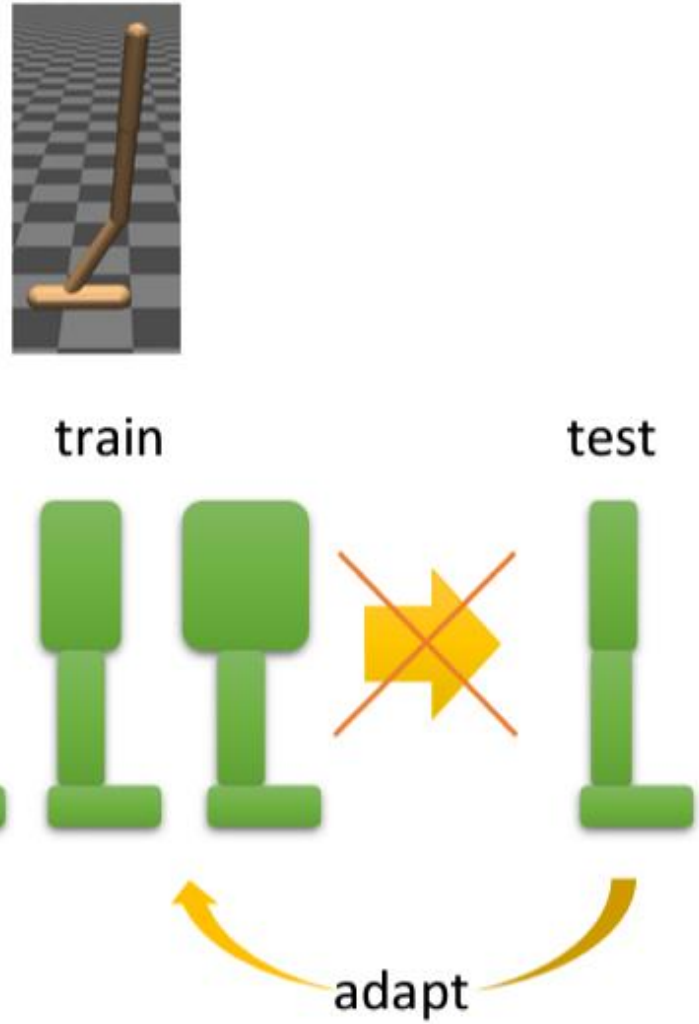
Pretraining with hierarchical RL methods:

Andreas et al. **Modular multitask reinforcement learning with policy sketches.** 2017.

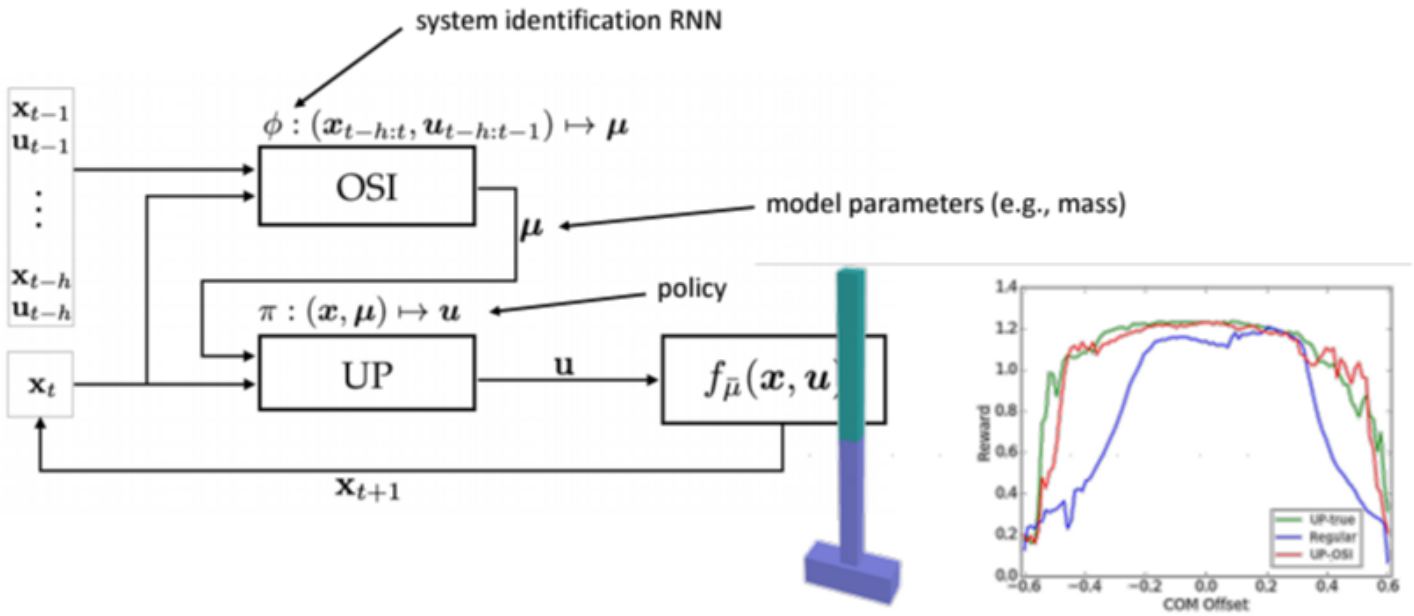
Florensa et al. **Stochastic neural networks for hierarchical reinforcement learning.** 2017.

¿Qué tal si podemos manipular el dominio origen?

- Podemos diseñar el dominio origen
- Simulación del mundo real
- La mayor diversidad en entrenamiento, mejor será la transferencia



Prepararse para lo desconocido



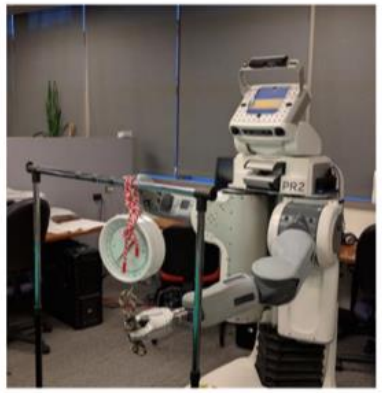
Yu et al., Preparing for the unknown: learning a universal policy with online system identification

CAD2RL: aleatoriedad para control en el mundo real

Sadeghi et al., CAD2RL:
real single-image flight
without a single real image

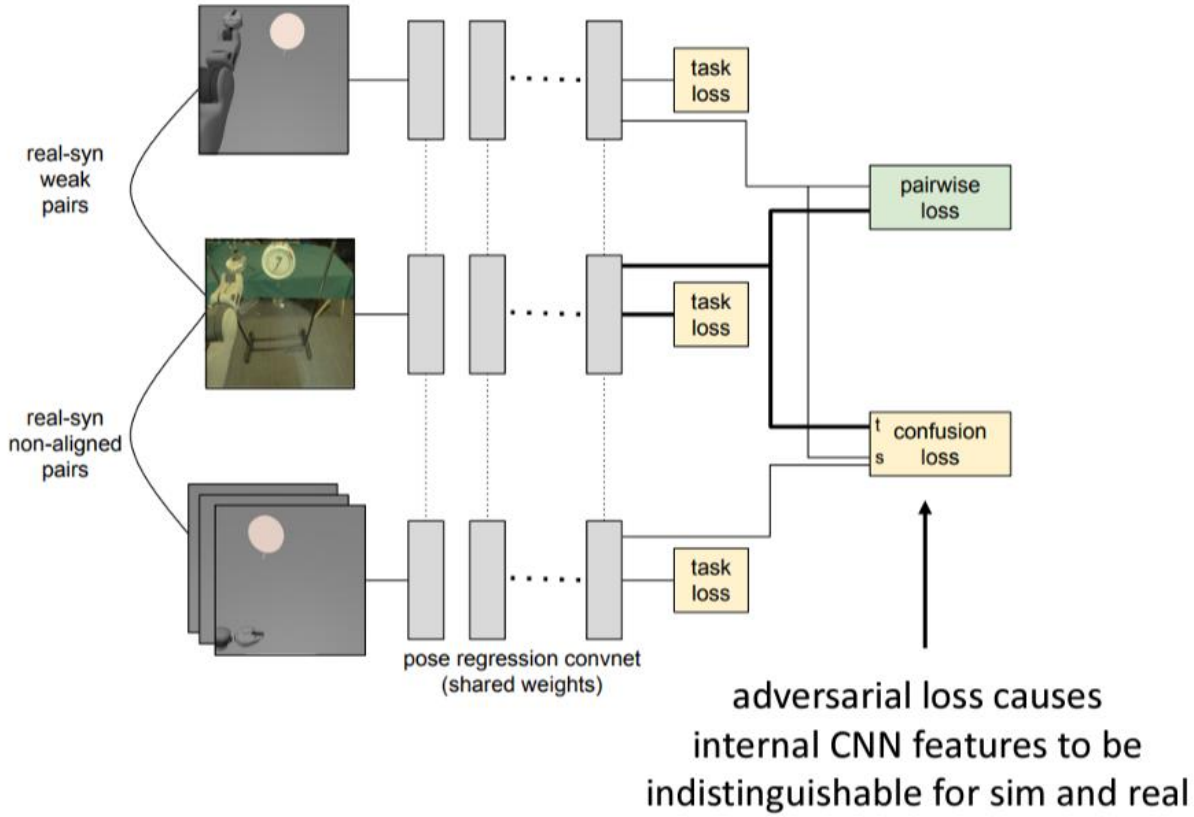
¿Qué tal si podemos observar el dominio objetivo?

Tzeng, Devin, et al. Adapting visuomotor representations with weak pairwise contraints



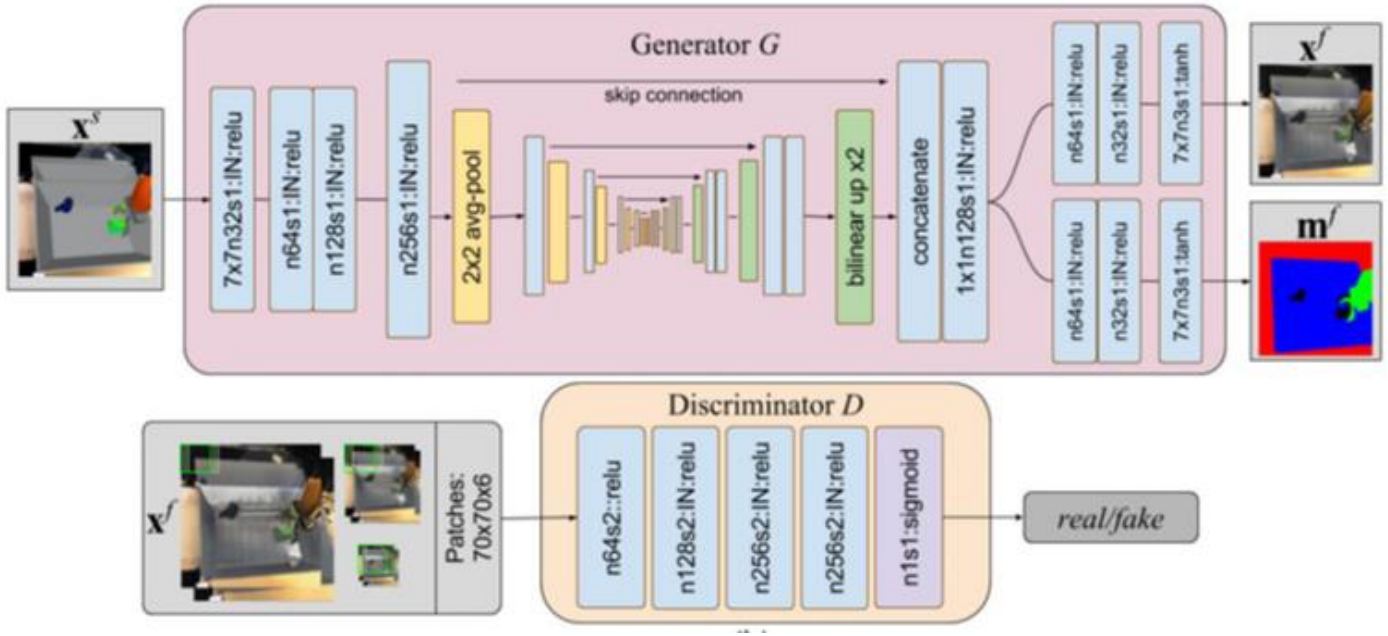
simulated images

real images



Adaptación de dominio a nivel de pixel

- Transformar una imagen sintética a una realista
- Bousmalis et al. Using simulation and domain adaptation to improve efficiency of Deep robotic grasping



Para saber más

Rajeswaran, et al. (2017). **EPOpt: Learning Robust Neural Network Policies Using Model Ensembles.**

Yu et al. (2017). **Preparing for the Unknown: Learning a Universal Policy with Online System Identification.**

Sadeghi & Levine. (2017). **CAD2RL: Real Single Image Flight without a Single Real Image.**

Tobin et al. (2017). **Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World.**

James et al. (2017). **Transferring End-to-End Visuomotor Control from Simulation to Real World for a Multi-Stage Task.**

Tzeng*, Devin*, et al. (2016). **Adapting Deep Visuomotor Representations with Weak Pairwise Constraints.**

Bousmalis et al. (2017). **Using Simulation and Domain Adaptation to Improve Efficiency of Deep Robotic Grasping.**

Dominios con múltiple origen

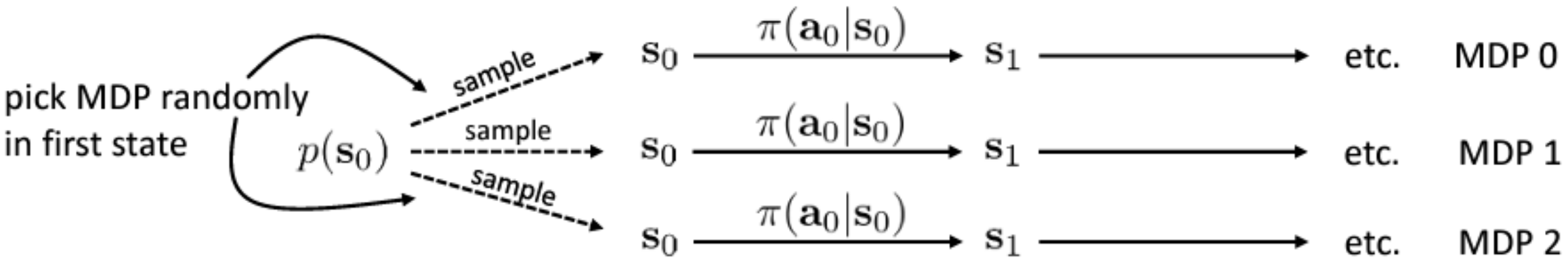
- Hasta ahora más diversidad es mejor transferencia
- Es necesario diseñar esa diversidad
- Es posible transferir de diferentes tareas
  - Más cercano a lo que hacen las personas: construir experiencia
  - Sustancialmente más difícil: las tareas pasadas no nos dicen como resolver la tarea en el dominio objetivo

RL basada en modelo

- Si las tareas anteriores son diferentes, ¿Qué tienen en común?
- Las leyes de la física
    - Mismo robot haciendo diferentes tareas
    - Mismo auto manejando a diferentes lugares
    - Tratar de hacer diferentes cosas en el mismo videojuego
- Versión simple
    - Entrenar el modelo en tareas pasadas y utilizarlo para nuevas tareas
- Versión compleja
    - Adaptar o ajuste fino del modelo a nueva tarea

¿Podemos resolver múltiples tareas?

- Algunos modelos son muy complicados
- Idea 1: construir un MDP con todas las tareas
- Idea 2: entrenar cada MDP separada y combinar sus politicas



Una política para todos los juegos de Atari

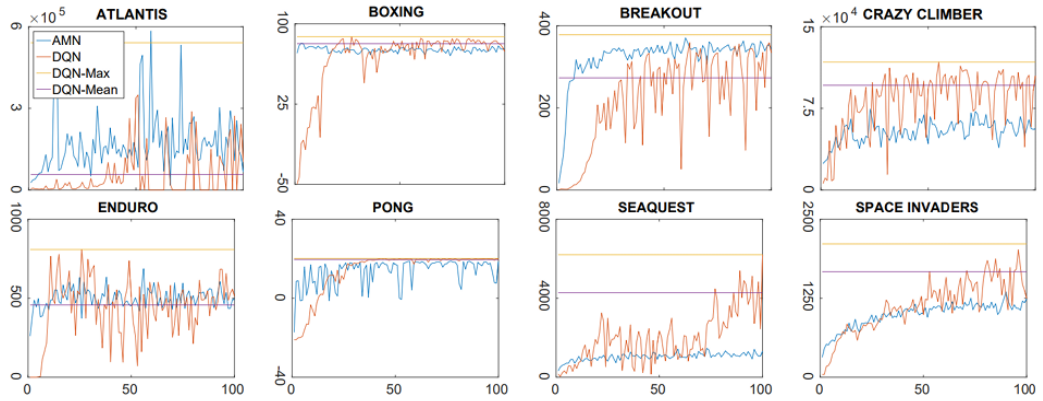
POLICY DISTILLATION

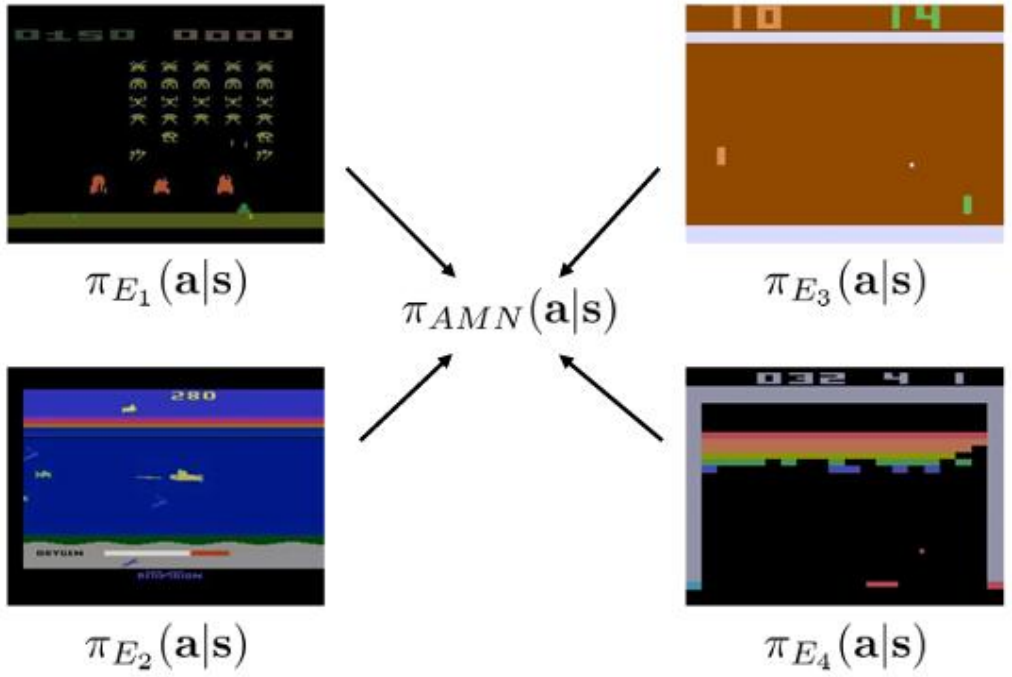
Andrei A. Rusu, Sergio Gómez Colmenarejo, Çağlar Gülçehre, Guillaume Desjardins,
James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu & Raia Hadsel
Google DeepMind

ACTOR-MIMIC DEEP MULTITASK AND TRANSFER REINFORCEMENT LEARNING

Emilio Parisotto, Jimmy Ba, Ruslan Salakhutdinov
Department of Computer Science
University of Toronto

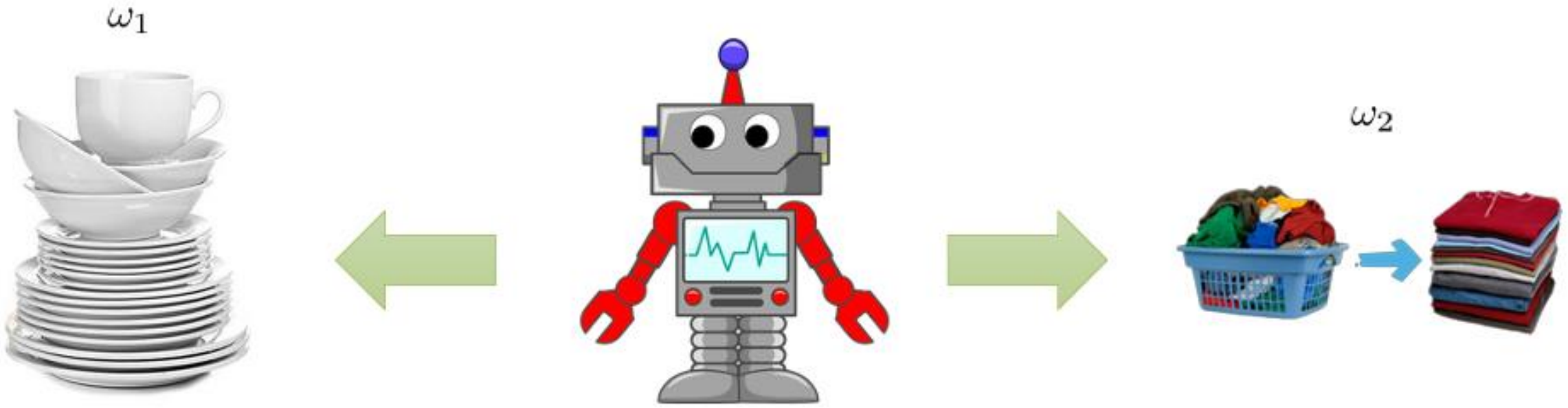
Destilación para transferencia multi-tarea





¿Cómo sabe un modelo que hacer?

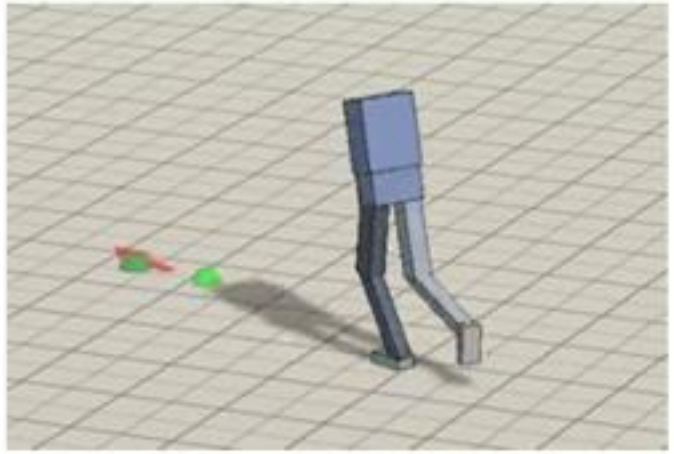
- Hasta ahora que hacer es aparente a partir de la entrada
- ¿Qué pasa si la política puede hacer múltiples cosas en el mismo ambiente?



Políticas contextuales

- Política estándar: $\pi_\theta(a|s)$
- Política contextual: $\pi_\theta(a|s,\omega)$
- Formalmente: $\tilde{s} = \begin{bmatrix} s \\ \omega \end{bmatrix}$ $\tilde{\mathcal{S}} = \mathcal{S} \times \Omega$

ω: stack location



ω: walking direction

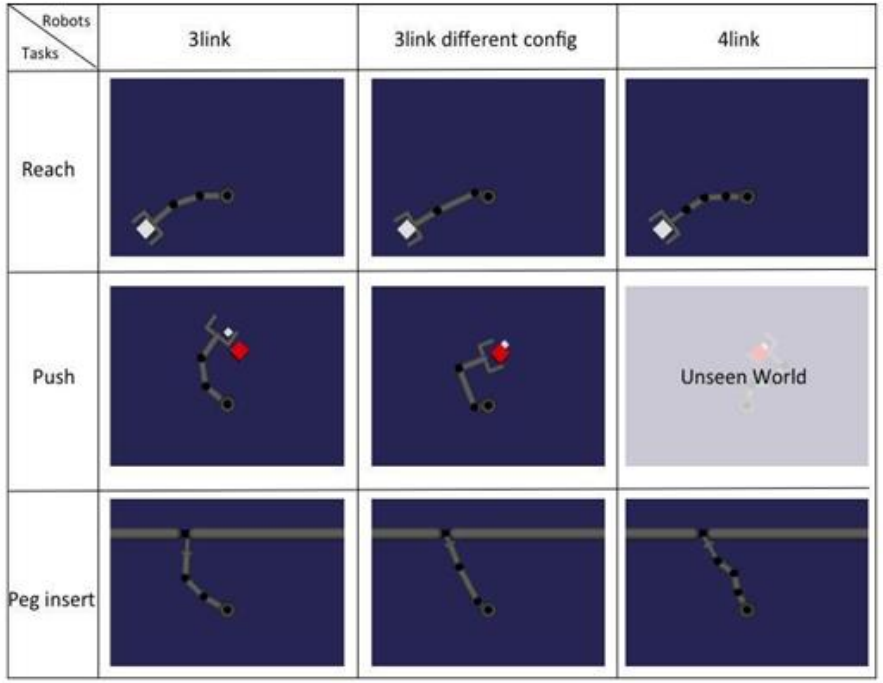
ω: where to hit puck

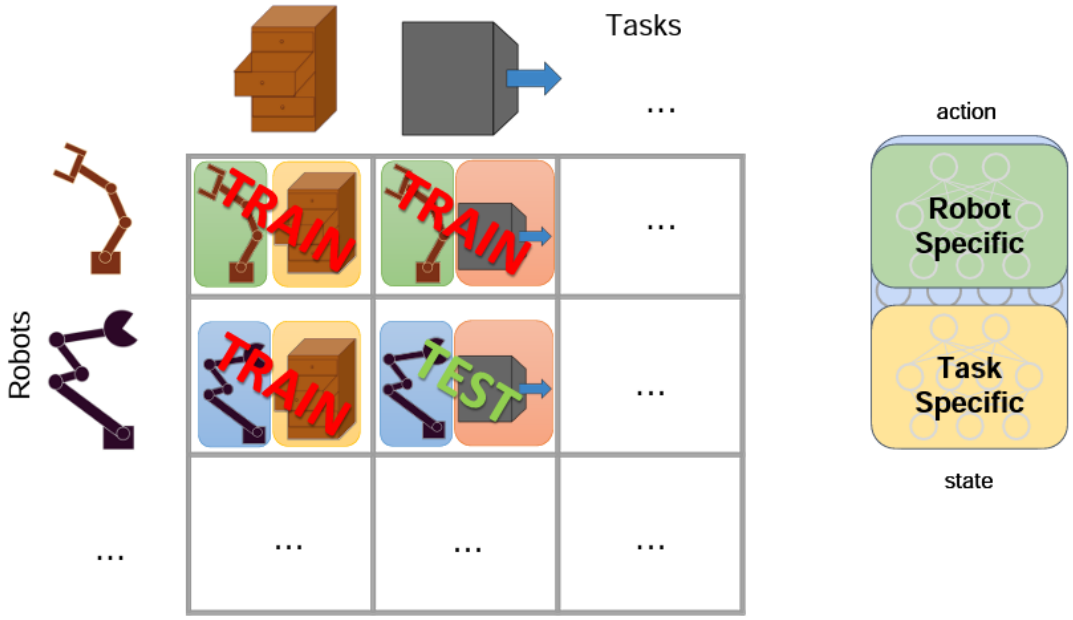
Arquitecturas para transferencia multitarea

- Hasta ahora, un modelo para todas las tareas
- Algunas tareas tienen partes compartidas
  - Dos autos iguales manejando en diferentes ciudades
  - Diez robots haciendo diferentes tareas
- Idea: realizar arquitecturas con componentes reutilizables

Redes modulares en RL

- Devin, Gupta, et al. Learning modular neural network policies





Para saber más

Fu etal. (2016). **One-Shot Learning of Manipulation Skills with Online Dynamics Adaptation and Neural Network Priors.**

Rusu et al. (2016). **Policy Distillation.**

Parisotto et al. (2016). **Actor-Mimic: Deep Multitask and Transfer Reinforcement Learning.**

Devin*, Gupta*, et al. (2017). **Learning Modular Neural Network Policies for Multi-Task and Multi-Robot Transfer.**

Para la otra vez…

- Meta aprendizaje