Transfer and Meta-Learning

~~Imitation and Inverse Reinforcement~~ Learning

- **Last time:**
  - Imitation Learning
  - Inverse Reinforcement Learning
- **Today:**

(Lecture content from Sergey Levine's CS 285 at Berkeley)

Imitation and Inverse Reinforcement Learning

- **Last time:**
  - Imitation Learning
  - Inverse Reinforcement Learning
- **Today:**
  - How do we **transfer** knowledge from one domain to another? (e.g. simulated to real-world)

(Lecture content from Sergey Levine's CS 285 at Berkeley)

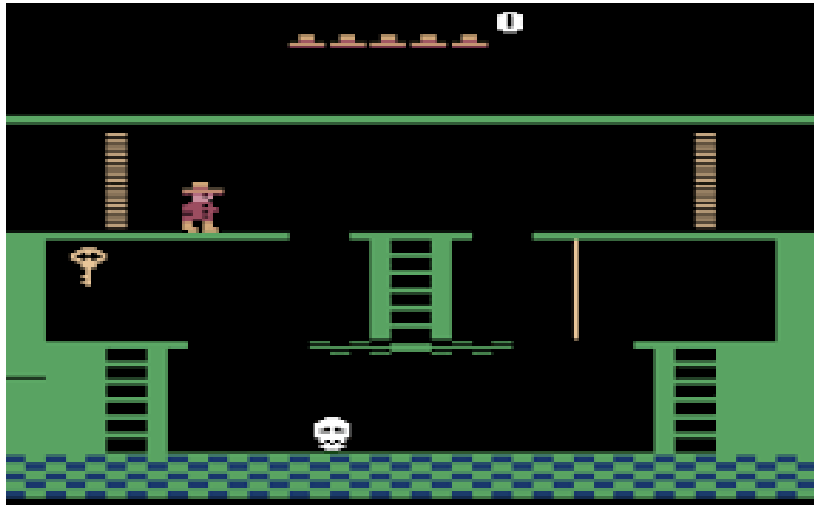
Imitation and Inverse Reinforcement Learning

- **Last time:**
  - Imitation Learning
  - Inverse Reinforcement Learning
- **Today:**
  - How do we **transfer** knowledge from one domain to another? (e.g. simulated to real-world)
  - How do we learn how to learn? (**Meta** learning)

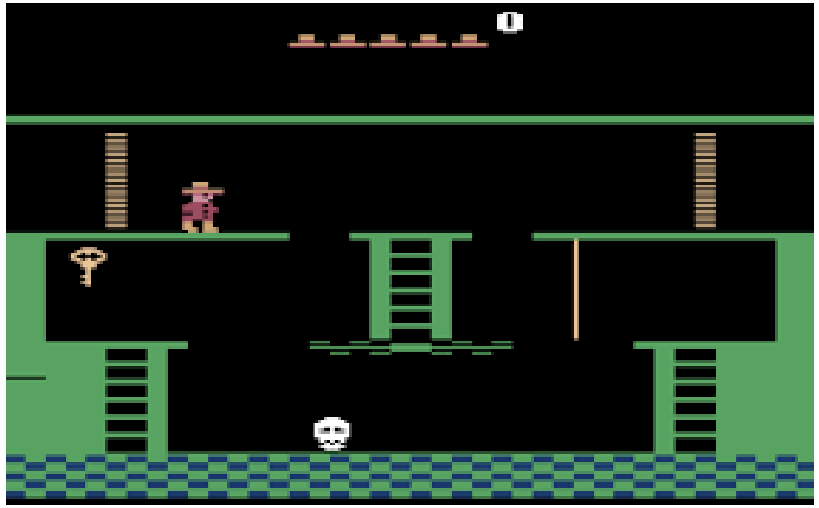
(Lecture content from Sergey Levine's CS 285 at Berkeley)

Transfer Learning and Montezuma's Revenge

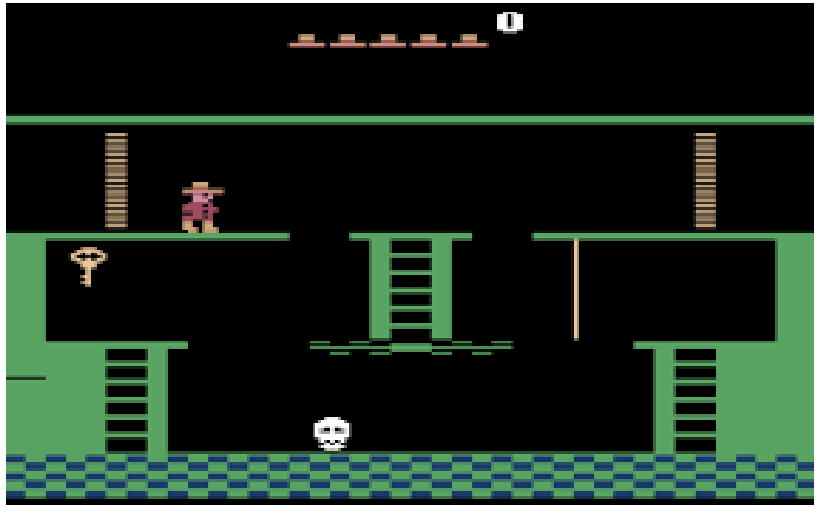
Transfer Learning and Montezuma's Revenge



Transfer Learning and Montezuma's Revenge



Transfer Learning and Montezuma's Revenge



Could an RL agent be better at Montezuma's revenge after watching Indiana Jones?

Transfer Learning

Transfer Learning

Transfer Learning: Use experience from one set of tasks for faster learning and better performance on a new task

Transfer Learning

Transfer Learning: Use experience from one set of tasks for faster learning and better performance on a new task

In RL, task=MDP

Transfer Learning

Transfer Learning: Use experience from one set of tasks for faster learning and better performance on a new task

In RL, task=MDP

Source domain $\rightarrow$ target domain

Transfer Learning

Transfer Learning: Use experience from one set of tasks for faster learning and better performance on a new task

In RL, task=MDP

Source domain $\rightarrow$ target domain

- "shot" = number of attempts in the target domain
- "0-shot" = run policy in target domain
- "1-shot" = try task once
- "few shot"

Transfer Learning

How should prior knowledge be stored?

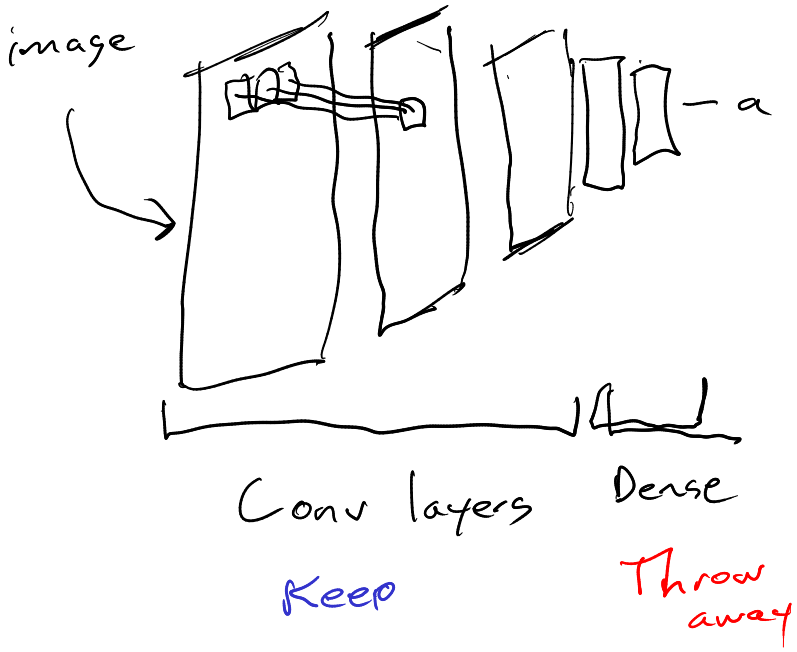
- Q-function
- Policy
- Model
- Features/hidden states

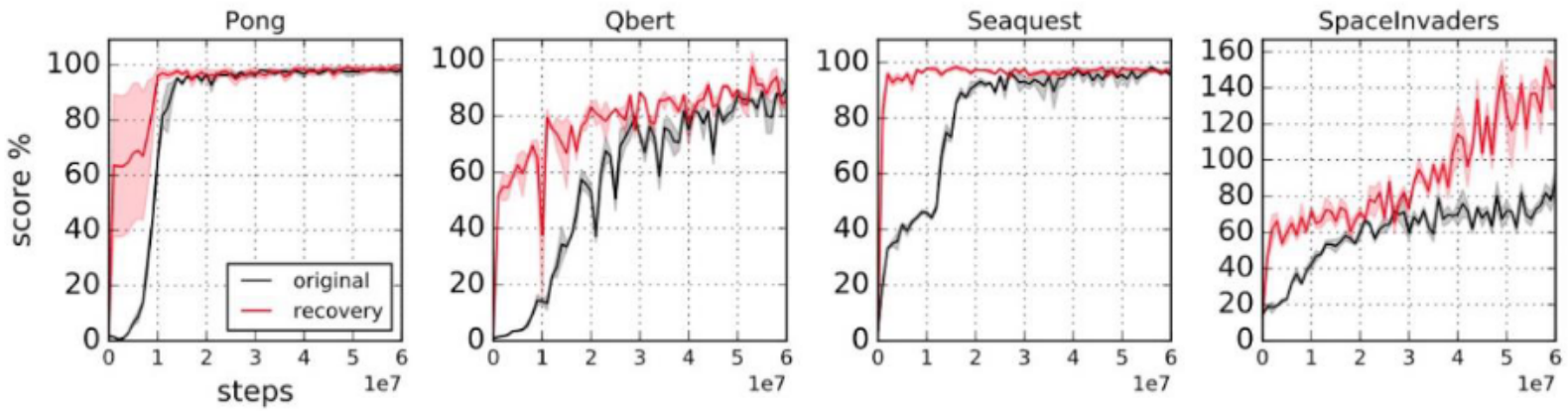
Transfer Learning

How should prior knowledge be stored?

- Q-function
- Policy
- Model
- Features/hidden states

Representation Bottleneck





Transfer Learning

How should prior knowledge be stored?

- Q-function
- Policy
- Model
- Features/hidden states

Transfer Learning

How should prior knowledge be stored?

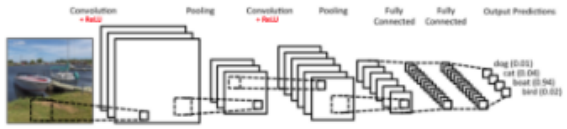
- Q-function
- Policy
- Model
- Features/hidden states

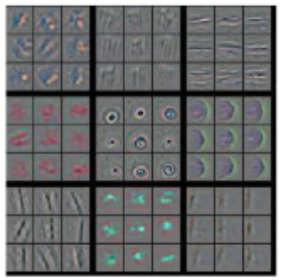
Pretraining + Finetuning

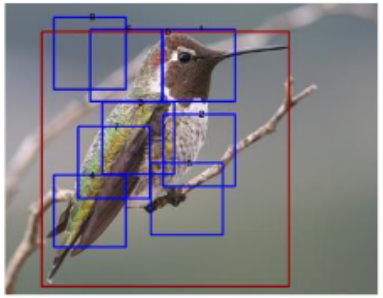
General Data

Domain-Specific Data

IMAGENET







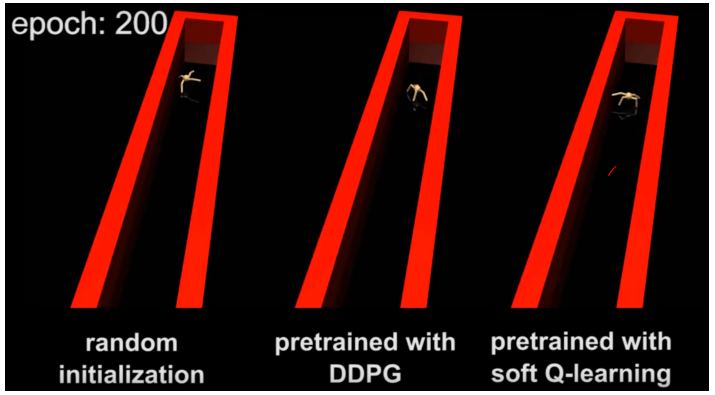
Pretraining + Finetuning

Pretraining + Finetuning

Pretrain: reward speed in any direction

Pretraining + Finetuning

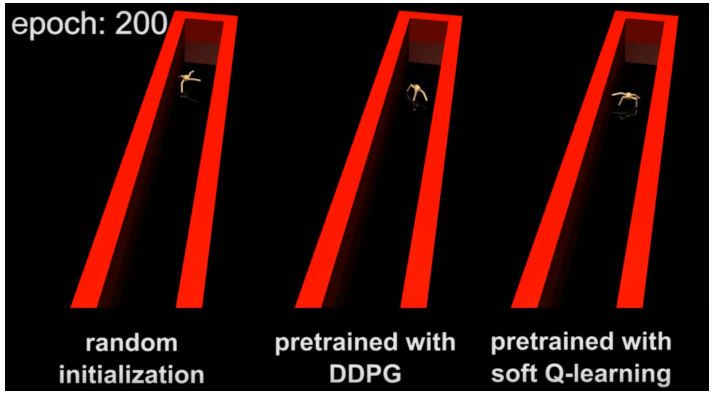
Pretrain: reward speed in any direction



Fine Tune: reward speed in specific direction

Pretraining + Finetuning

Pretrain: reward speed in any direction



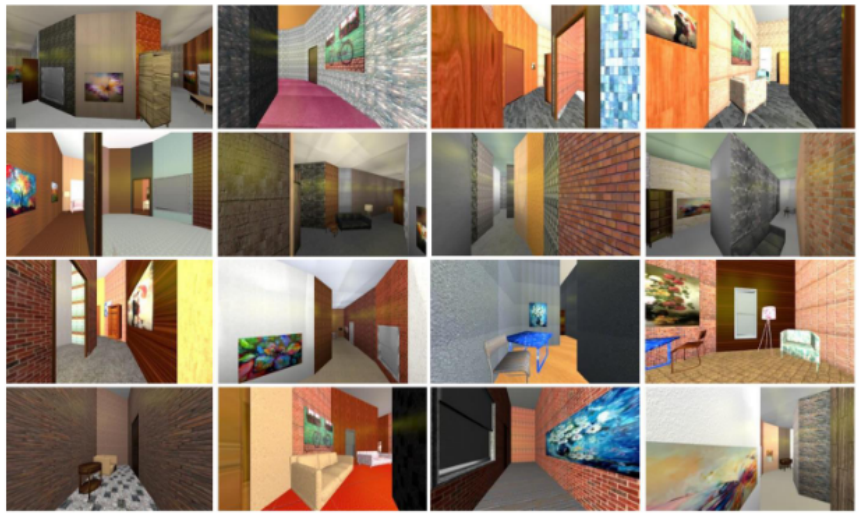
Fine Tune: reward speed in specific direction

softmax policy

$$\pi(a|s) \propto_a \exp(Q(s,a))$$

https://sites.google.com/view/softqlearning/home

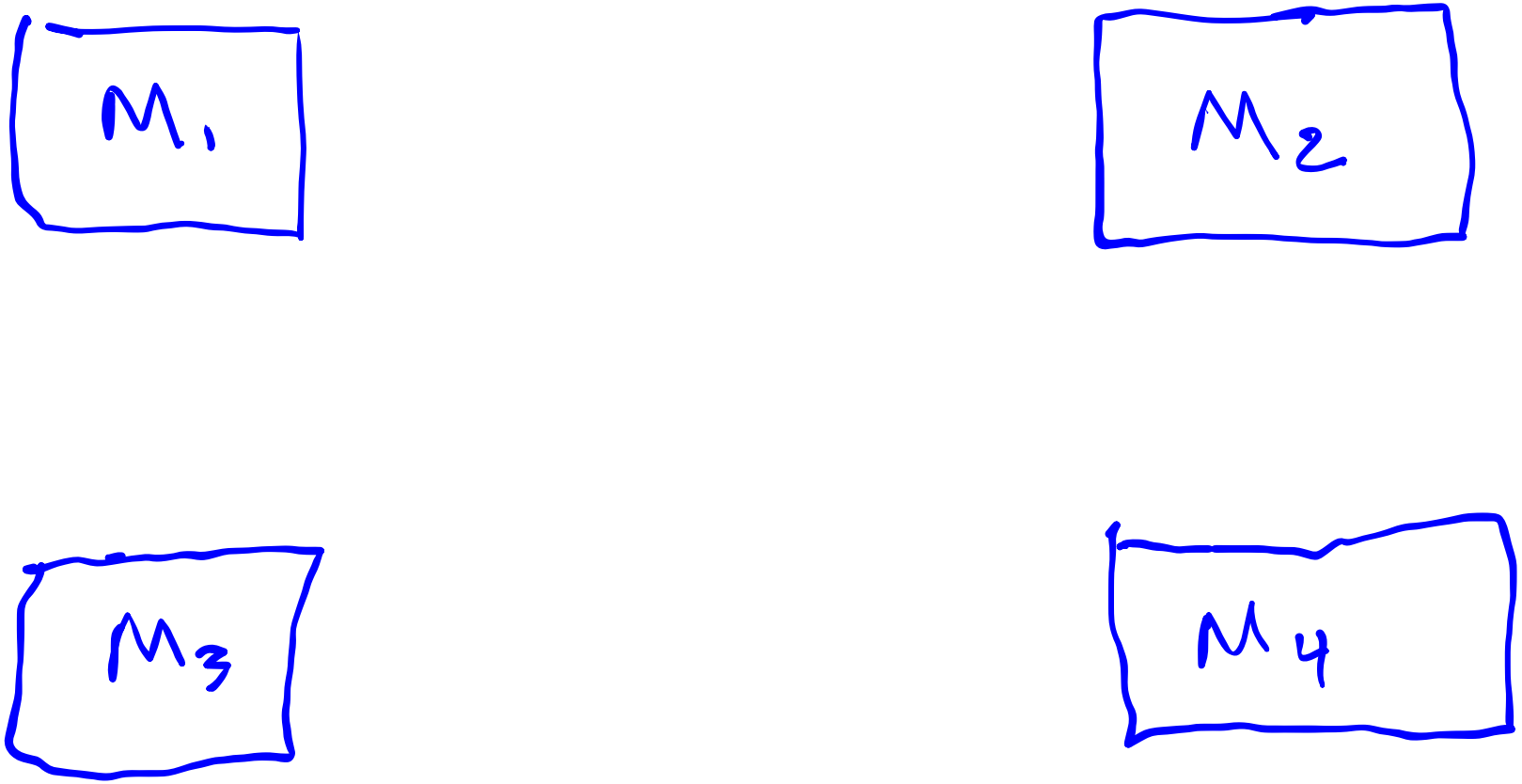
CAD2RL



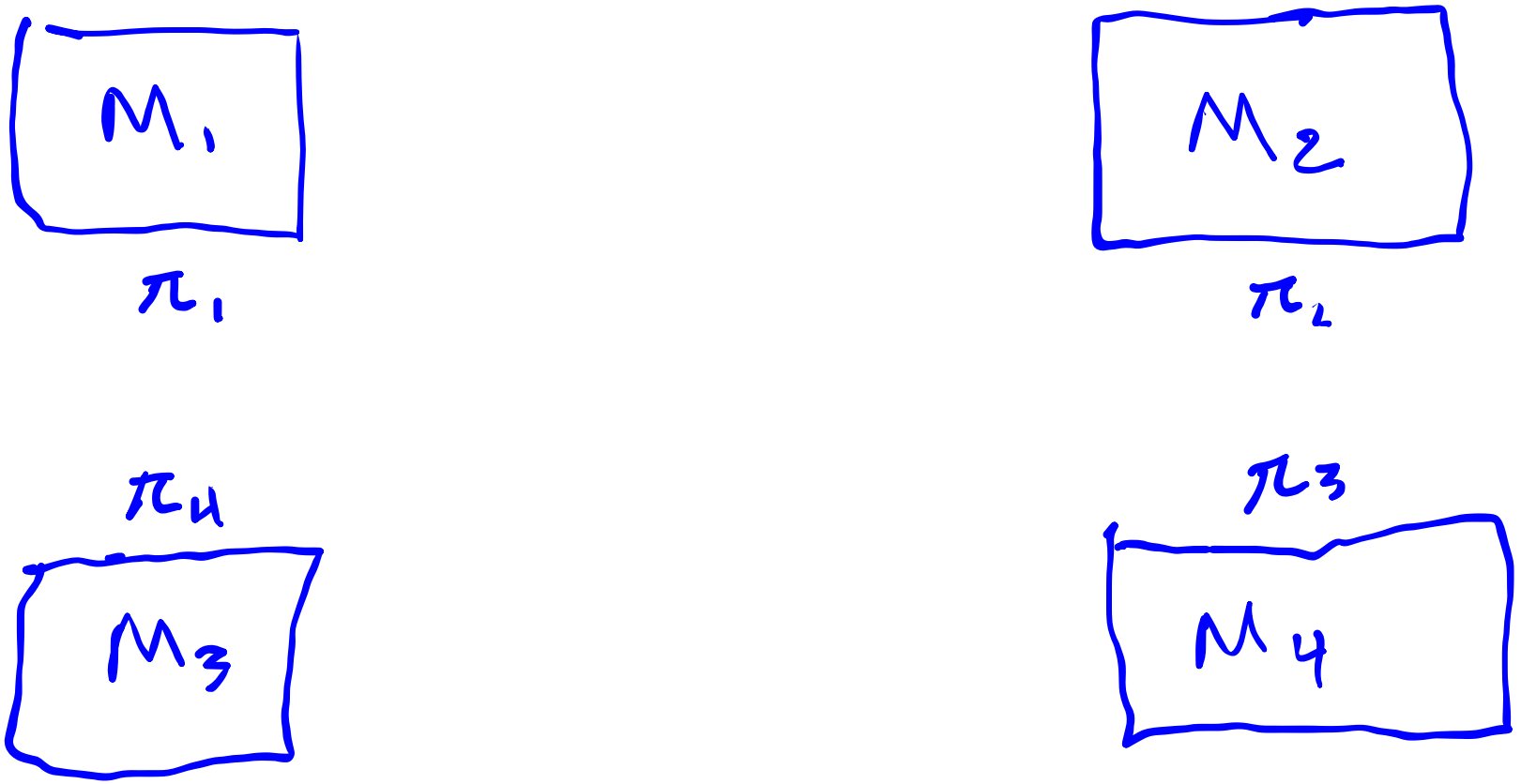
Key: Diversity

Actor Mimic

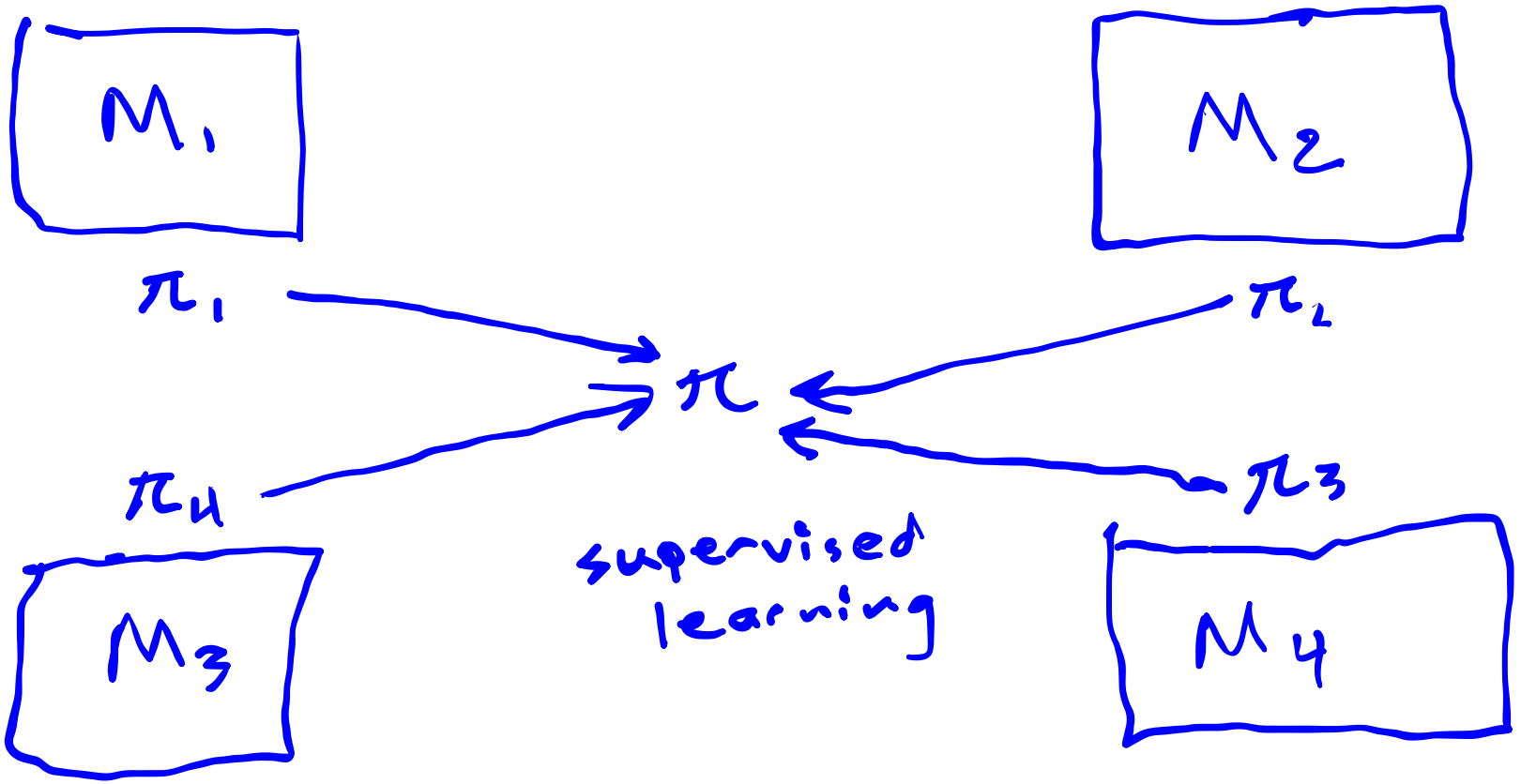
Actor Mimic



Actor Mimic



Actor Mimic



Transfer Learning

How should prior knowledge be stored?

- Q-function
- Policy
- Model
- Features/hidden states

Transfer Learning

How should prior knowledge be stored?

- Q-function
- Policy
- Model
- Features/hidden states

Successor Features

All domains have same S, A, T, γ

Difference: R

Recall:

$$V^{\pi} = (I - \gamma T^{\pi})^{-1} R^{\pi}$$

Let \(R(s, a) =