Last Time: DQN with images

This lecture: Can we use model-based methods with images?
Recap: model-based RL

model-based reinforcement learning version 1.0:

1. run base policy $\pi_0(u_t|x_t)$ (e.g., random policy) to collect $D = \{(x, u, x')_i\}$

2. learn dynamics model $f(x, u)$ to minimize $\sum_i \|f(x_i, u_i) - x'_i\|^2$

3. backpropagate through $f(x, u)$ to choose actions (e.g. using iLQR)

4. execute those actions and add the resulting data $\{(x, u, x')_j\}$ to $D$

What about POMDPs?
Outline

1. Models in latent space
2. Models directly in image space
3. Inverse models

Note: This is an active area of research.
Outline

1. Models in latent space
2. Models directly in image space
3. Inverse models
Learning in Latent Space

Key idea: learn embedding $g(o_t)$, then learn in latent space (model-based or model-free)

What do we want $g$ to be?
It depends on the method — we’ll see.
Learning in Latent Space

**Key idea**: learn embedding \( g(o_t) = x_t \), then learn in latent space (model-based or **model-free**)

**Autonomous reinforcement learning on raw visual input data in a real world application**

Sascha Lange, Martin Riedmiller
Department of Computer Science
Albert-Ludwigs-Universität Freiburg

Arne Voigtlander
Shogee GmbH & Co. KG
Krügerweg 16a

controlling a slot-car
1. collect data with exploratory policy
2. learn **low-dimensional** embedding of image (how?)
3. run q-learning with function approximation with embedding

embedding is **low-dimensional** and summarizes the image
1. collect data with exploratory policy
2. learn **low-dimensional** embedding of image (how?)
3. run q-learning with function approximation with embedding

**Pros:**
+ Learn visual skill very efficiently

**Cons:**
- Autoencoder might not recover the right representation
- Not necessarily suitable for model-based methods
Learning in Latent Space

Key idea: learn embedding $g(o_t) = x_t$, then learn in latent space (model-based or model-free)

Deep Spatial Autoencoders for Visuomotor Learning

Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, Pieter Abbeel

Fig. 1: PR2 learning to scoop a bag of rice into a bowl with a spatula (left) using a learned visual state representation (right).
1. collect data with exploratory policy
2. learn **smooth, structured** embedding of image
3. learn local-linear model with embedding
4. run iLQG to learn to reach image of goal & goal gripper pose

embedding is **smooth and structured**
1. collect data with exploratory policy
2. learn **smooth, structured** embedding of image
3. learn local-linear model with embedding
4. run iLQG to learn to reach **image of goal** & goal gripper pose

Because we aren't using states, we need a reward.
autonomous execution 6x real-time
Our Method
autonomous execution
real-time
Our Method
autonomous execution

real-time
O - current feature point
X - goal feature point

autonomous execution
1. collect data with exploratory policy
2. learn **smooth, structured** embedding of image
3. learn local-linear model with embedding
4. run iLQG to learn to reach image of goal & goal gripper pose

**Pros:**

+ Learn complex visual skill very efficiently
+ Structured representation enables effective learning

**Cons:**

- Autoencoder might not recover the right representation
Learning in Latent Space

**Key idea**: learn embedding $g(o_t) = x_t$, then learn in latent space (model-based or model-free)

---

**Embed to Control: A Locally Linear Latent Dynamics Model for Control from Raw Images**

Manuel Watter*  
Jost Tobias Springenberg*  
Joschka Boedecker  
University of Freiburg, Germany  
{watterm, springj, jboedeck}@cs.uni-freiburg.de

Martin Riedmiller  
Google DeepMind  
London, UK  
riedmiller@google.com
1. collect data
2. learn embedding of image & dynamics model (jointly)
3. run iLQG to learn to reach image of goal

embedding that can be modeled
Swing-up with the E2C algorithm
Thought exercise:
Why reconstruct the image?
Why not just learn embedding and model on embedding?
Outline

1. Models in latent space
2. Models directly in image space
3. Inverse models
Models with Images

Action-conditioned video prediction $f(o_t, u_t) = o_{t+1}$

Action-Conditional Video Prediction using Deep Networks in Atari Games

Junhyuk Oh  Xiaoxiao Guo  Honglak Lee  Richard Lewis  Satinder Singh
University of Michigan, Ann Arbor, MI 48109, USA

(a) Feedforward encoding
(b) Recurrent encoding
Models with Images

Action-conditioned video prediction \( f(o_t, u_t) = o_{t+1} \)

Key components:
multi-step prediction \( f(o_t, u_{t:T-1}) = o_{t+1:T} \)
curriculum learning and/or scheduled sampling
Does it work? Yes!
can make 100-step predictions
Does it work?  Maybe not.

fails to model a critical part of the game
Does it work?

(a) Seaquest  (b) Space Invaders  (c) Freeway  (d) QBert  (e) Ms Pacman

Figure 3: Mean squared error over 100-step predictions
Is it useful?

Using model for informed exploration
Using model for informed exploration:

1. Store most recent $d$ frames
2. For every valid action, predict 1 frame ahead
3. Take action corresponding to future frame least like the previous $d$ frames

Use Gaussian kernel similarity metric on images:

$$n_D(x^{(a)}) = \sum_{i=1}^{d} k(x^{(a)}, x^{(i)}); \quad k(x, y) = \exp\left(-\sum_{j} \min(\max((x_j - y_j)^2 - \delta, 0), 1)/\sigma\right)$$

*caveat*: prediction model was trained with data from DQN agent

more on exploration later in this course!
Pros:
+ Stability through multi-step prediction
+ Useful for control

Cons:
- Synthetic images are easier to generate
- Not immediately clear how to plan with it

Action-conditioned video prediction: $f(o_t, u_t) = o_{t+1}$
What about real images?

Unsupervised Learning for Physical Interaction through Video Prediction

Chelsea Finn* 
UC Berkeley

Ian Goodfellow 
OpenAI

Sergey Levine 
Google Brain

Deep Visual Foresight for Planning Robot Motion

Chelsea Finn¹,² and Sergey Levine¹,²
Data collection - 50k sequences (1M+ frames)

data publicly available for download sites.google.com/site/brainrobotdata
Train 8-step predictive model

Atari recurrent model

evaluate on held-out objects

→ doesn’t have capacity to represent real images.
Train predictive model

**action-conditioned** multi-frame video prediction

via **flow prediction**

- feed back model’s predictions for multi-frame prediction
- trained with $l_2$ loss
Train predictive model

**convolutional LSTMs**

- Action-conditioned
- Stochastic flow prediction
- Evaluate on held-out objects
Train predictive model

Finn et al., ‘16

Kalchbrenner et al., ‘16

Are these predictions good? accurate? useful?
What is prediction good for?

action magnitude: 0x  0.5x  1x  1.5x
Planning with Visual Foresight (MPC)

1. Sample N potential action sequences
2. Predict the future for each action sequence
3. Pick best future & execute corresponding action
4. Repeat 1-3 to replan in real time
Which future is the best one?

Specify goal by selecting where pixels should move.

Select future with maximal probability of pixels reaching their respective goals.
How it works
Does it work?

Only human involvement during training is: programming initial motions and providing objects to play with.

- evaluation on short pushes of novel objects
- translation & rotation
action-conditioned multi-frame video prediction via flow prediction

Pros:
+ Real images
+ Very limited human involvement (self-supervised)
+ Approach should improve as video prediction methods improve

Cons:
- Despite real images, limited background variability
- Somewhat simple skills
- Compute intensive at test-time
Outline

1. Models in latent space
2. Models directly in image space
3. Inverse models
Inverse Models

Thought exercise revisited:
Why reconstruct the image?

Learn embedding via inverse model $f(o_t, o_{t+1}) = u_t$
Inverse Models

Learn embedding via inverse model $f(o_t, o_{t+1}) = u_t$

---

Learning to Poke by Poking: Experiential Learning of Intuitive Physics

---

Pulkit Agrawal*  Ashvin Nair*  Pieter Abbeel  Jitendra Malik  Sergey Levine
Berkeley Artificial Intelligence Research Laboratory (BAIR)
University of California Berkeley
Learn embedding via inverse model \( f(o_t, o_{t+1}) = u_t \)

**regularize** embedding with forward model
Learn embedding via inverse model $f(o_t, o_{t+1}) = u_t$

Greedily plan with inverse model and image of goal
Qualitative Results

[Images showing initial, final, and target states for objects labeled A and B]
Learn embedding via inverse model \( f(o_t, o_{t+1}) = u_t \)

**Pros:**
+ Very limited human involvement (self-supervised)
+ Don’t have to reconstruct image

**Cons:**
- Can’t plan with inverse model
- Inverse model objective just cares about action
Model-Based vs. Model-Free Learning

Models:
+ Easy to collect data in a scalable way (self-supervised)
+ Possibility to transfer across tasks
+ Typically require a smaller quantity of supervised data
- Models don’t optimize for task performance
- Sometimes harder to learn than a policy
- Often need assumptions to learn complex skills (continuity, resets)

Model-Free:
+ Makes little assumptions beyond a reward function
+ Effective for learning complex policies
- Require a lot of experience (slower)
- Not transferable across tasks
Advanced Model Learning Takeaways

- Learning the **right** features is important

- Need to think about reward/objective when using models of observations

**Next time:** advanced imitation learning