Carnegie Mellon

School of Computer Science

Deep Reinforcement Learning and Control

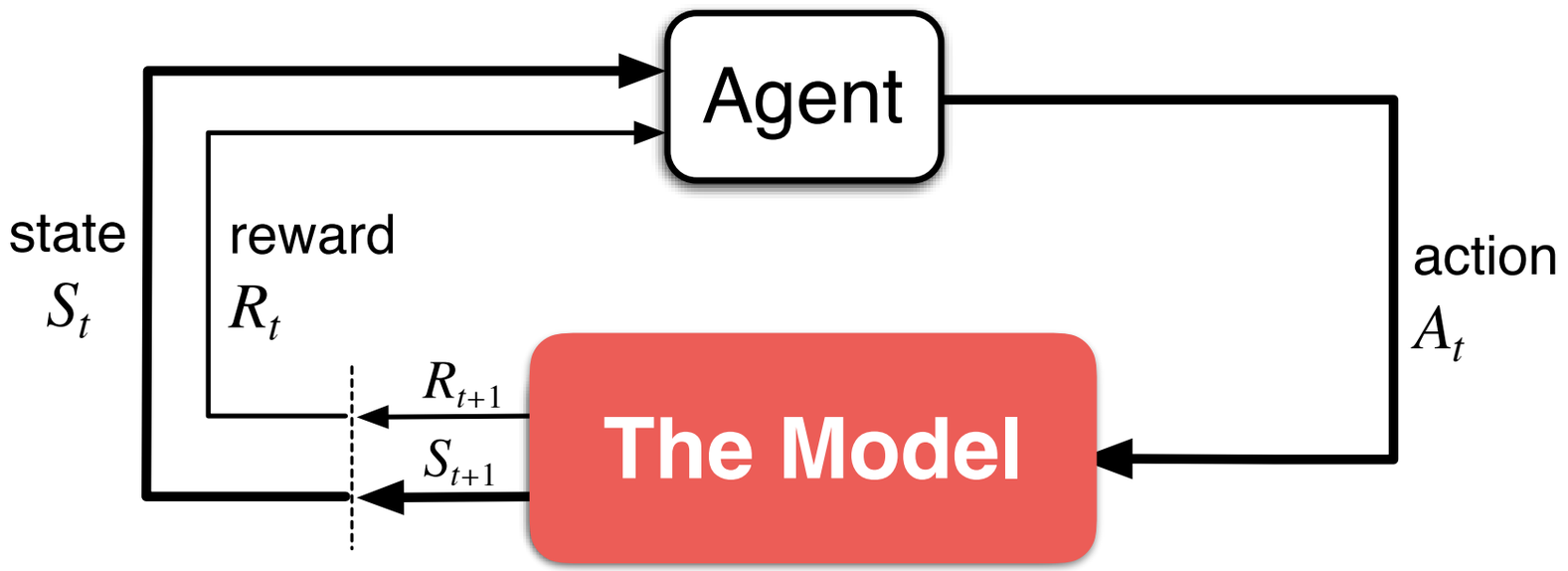
Model Based Reinforcement Learning in the sensory space

Spring 2020, CMU 10-403

Katerina Fragkiadaki

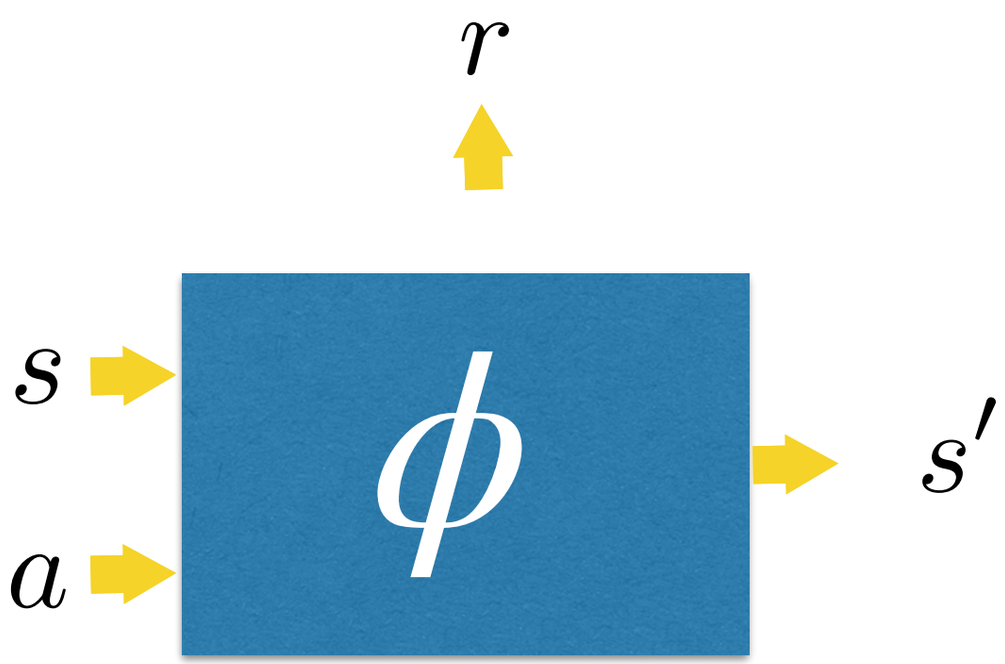
Planning

Planning: any computational process that uses a model to create or improve a policy



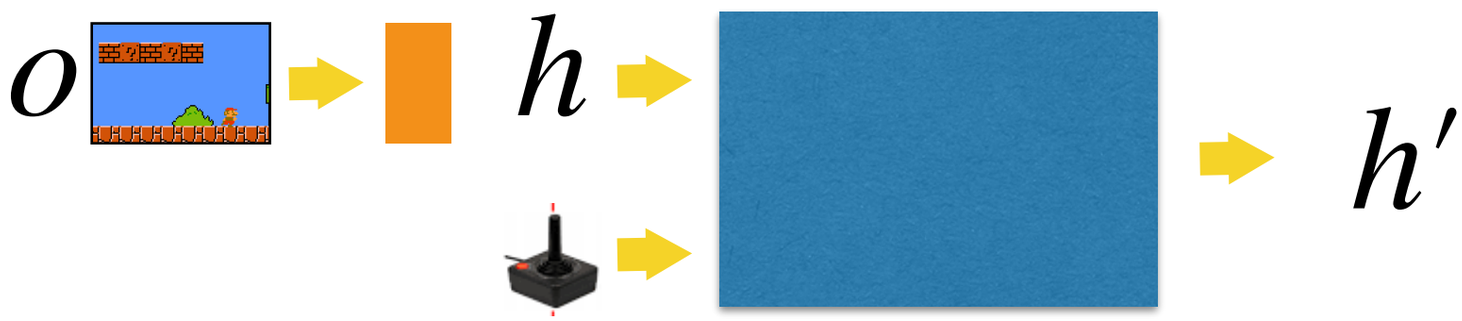
Model learning

We will be learning the model using experience tuples. A supervised learning problem.



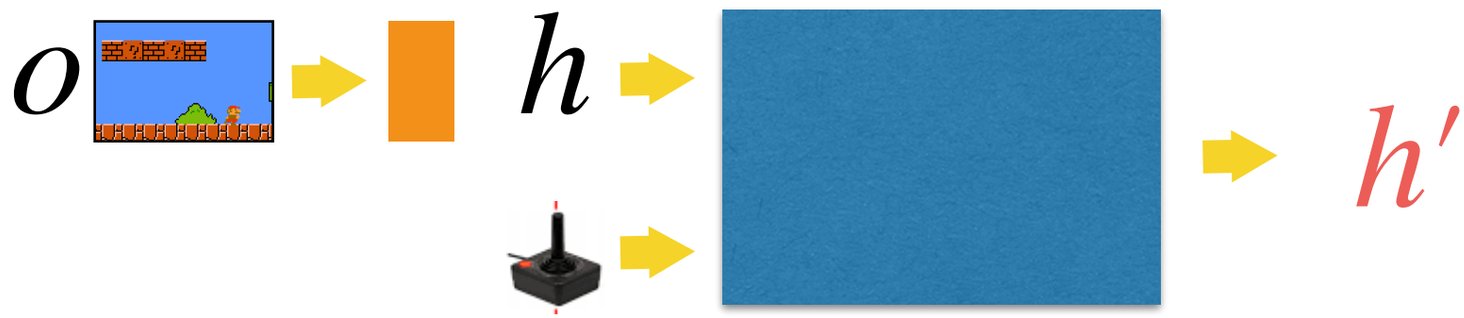
gaussian process,
random forest, deep
neural network,
linear function

Model Learning - 3 Qs always in mind

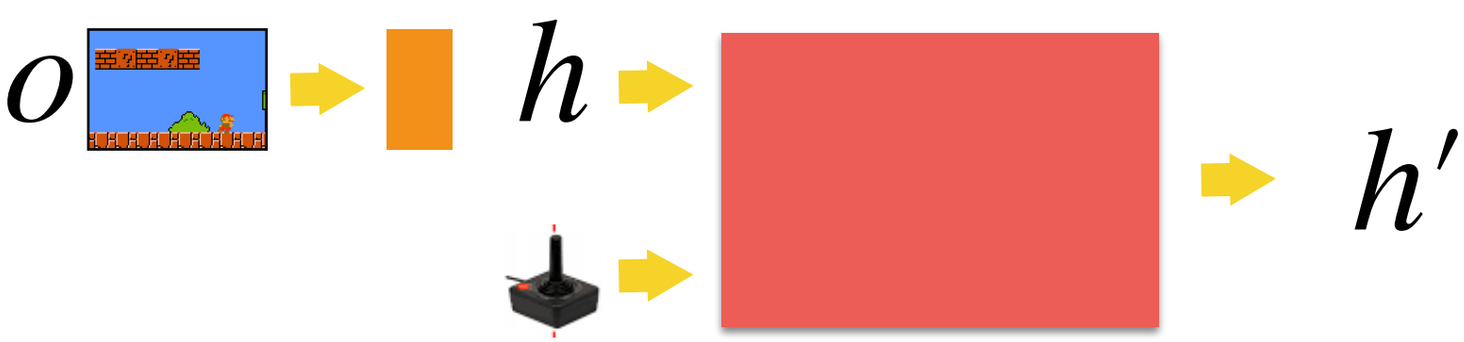


Model Learning - 3 Qs always in mind

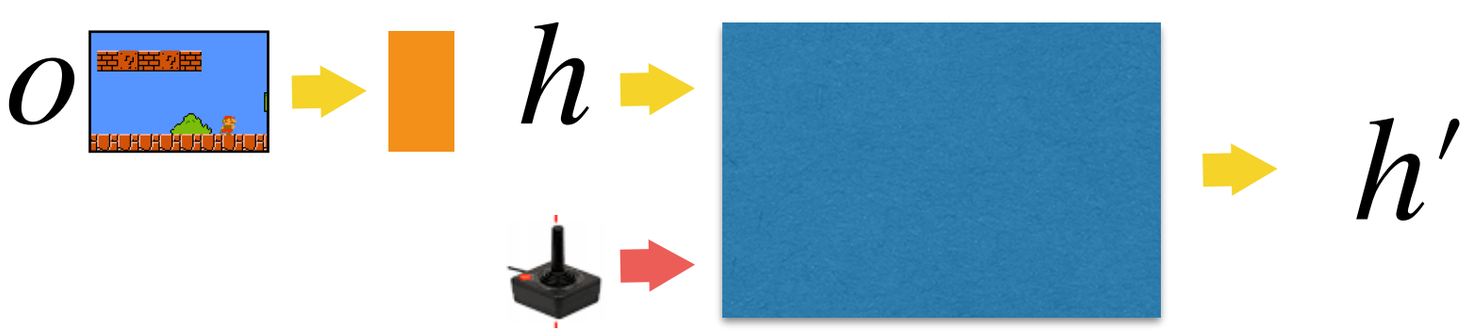
- What shall we be predicting?



- What is the architecture of the model, what structural biases should we add to get it to generalize?

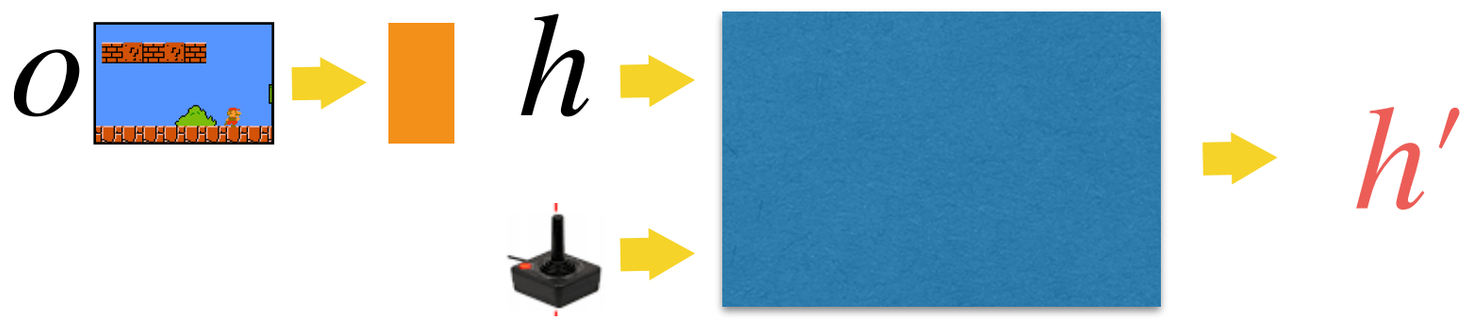


- What is the action representation?



Model Learning - 3 Qs always in mind

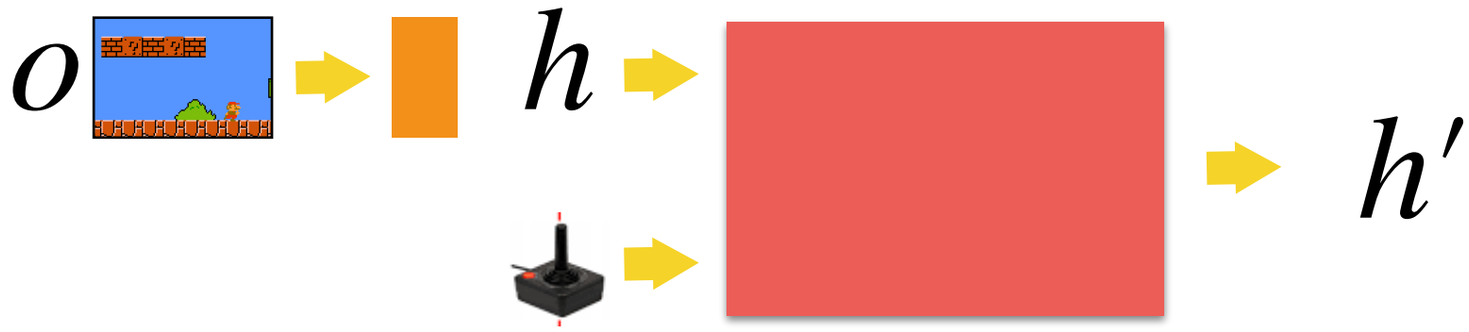
- What shall we be predicting?



Answers:
1.Future images (WXHX3 matrices)
2.Rewards
3.Object motions (in 2D)
4.Image abstractions
5.Object abstractions

Model Learning - 3 Qs always in mind

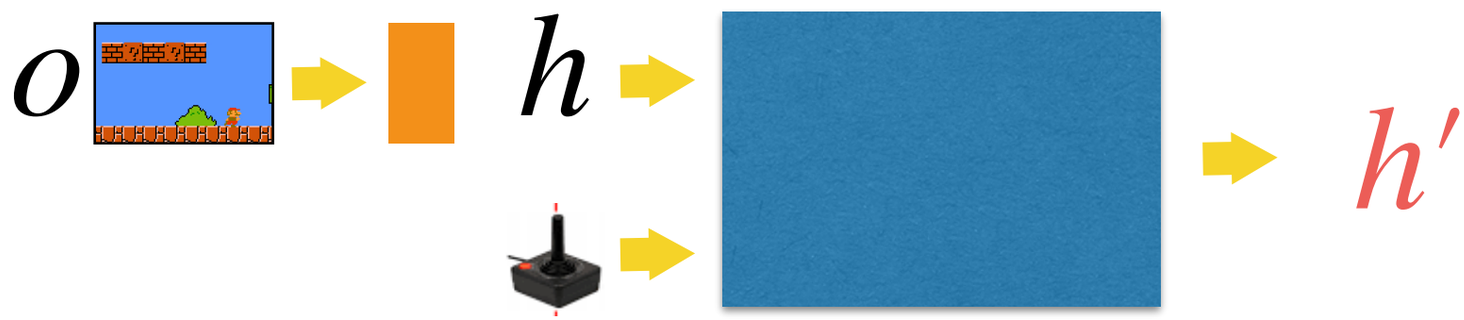
- What is the architecture of the model, what structural biases should we add to get it to generalize?



Answers:
1.CNNs
2.Object-centric CNNs
3.Graph neural nets

Model Learning - 3 Qs always in mind

- What shall we be predicting?



Answers:

1.Future images (WXHX3 matrices)

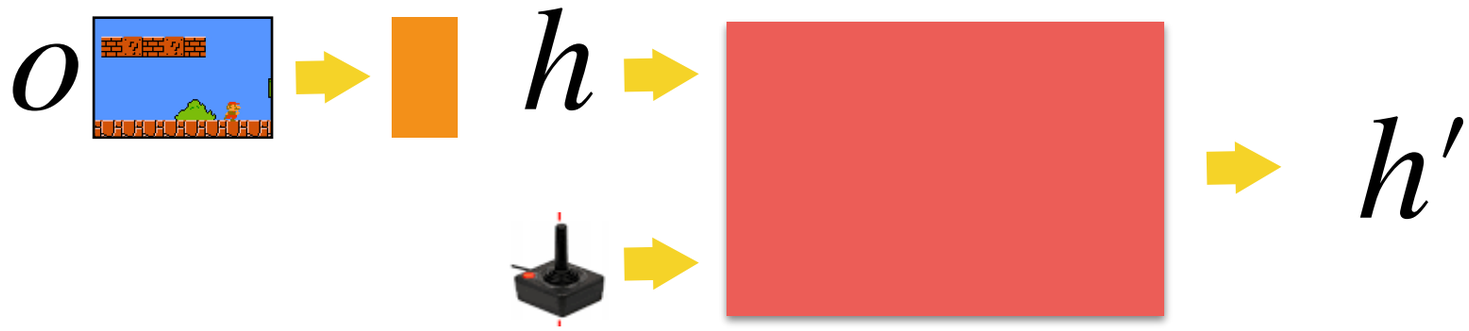
2.Object motions (in 2D)

3.Image abstractions

4.Object abstractions

Model Learning - 3 Qs always in mind

- What is the architecture of the model, what structural biases should we add to get it to generalize?



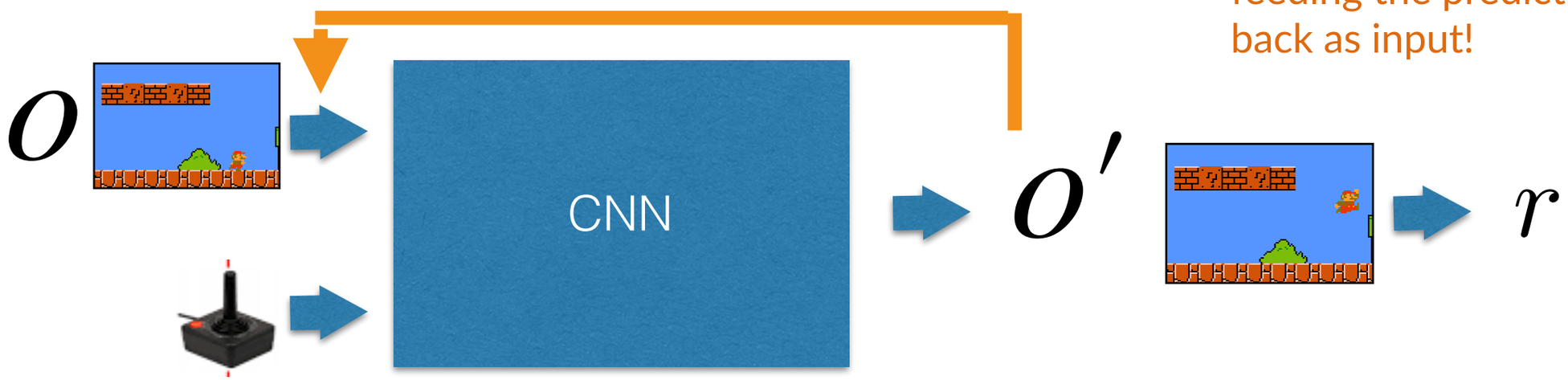
Answers:

1.CNNs

2.Object-centric CNNs

3.Graph neural nets

Model learning in image space (a.k.a. video prediction)



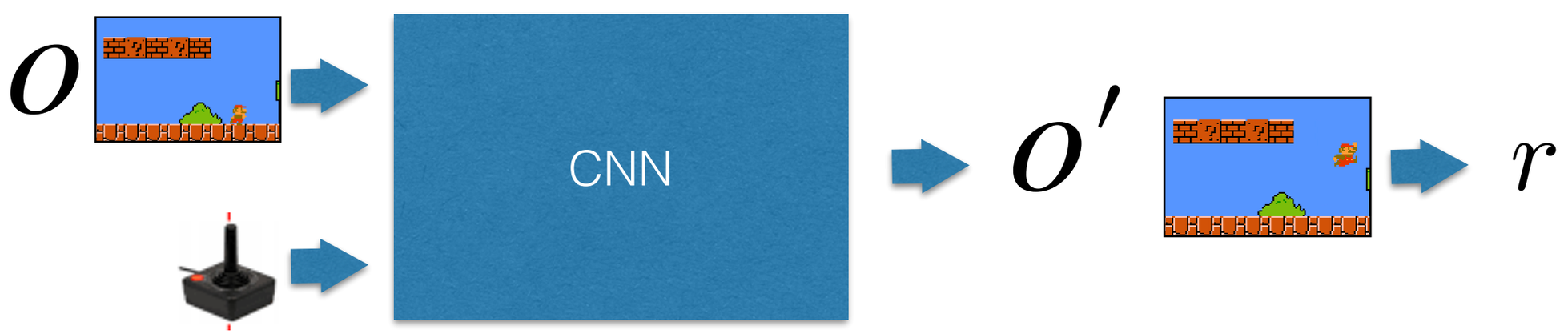
Unroll the model by feeding the prediction back as input!

MANY different rewards can be computed from the future visual observation, e.g., make Mario jump, make Mario move to the right, to the left, lie down, make Mario jump on the well and then jump back down again etc..

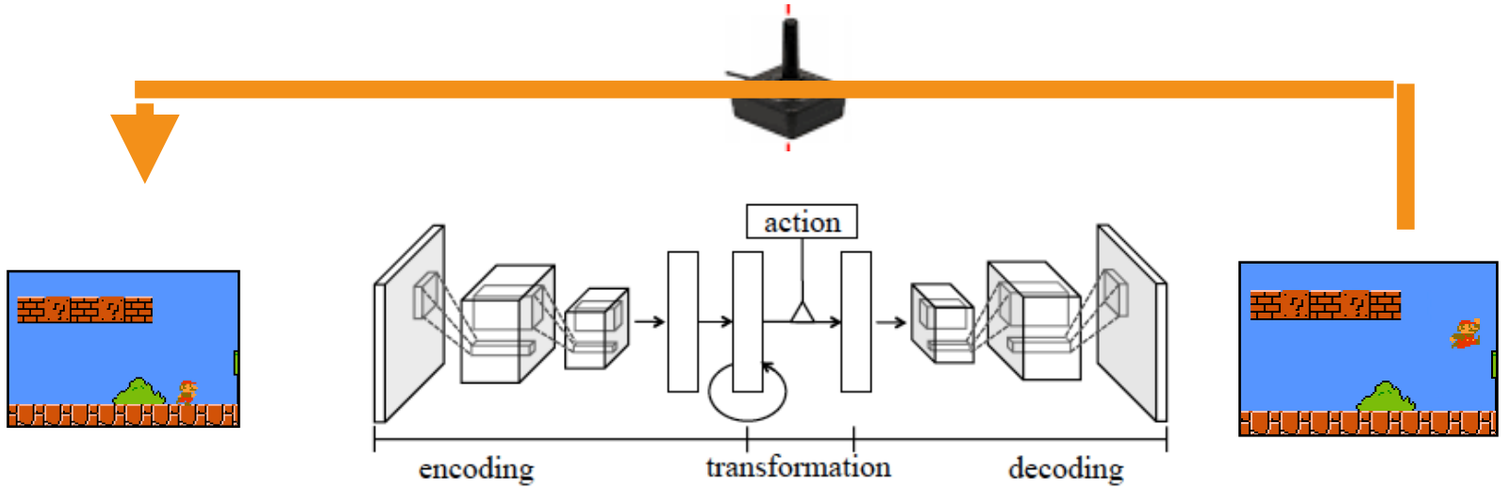
Action-Conditional Video Prediction using Deep Networks in Atari Games

Junhyuk Oh Xiaoxiao Guo Honglak Lee Richard Lewis Satinder Singh

- Train a neural network that given an image (sequence) and an action, predict the pixels of the next frame
- Unroll it forward in time to predict multiple future frames



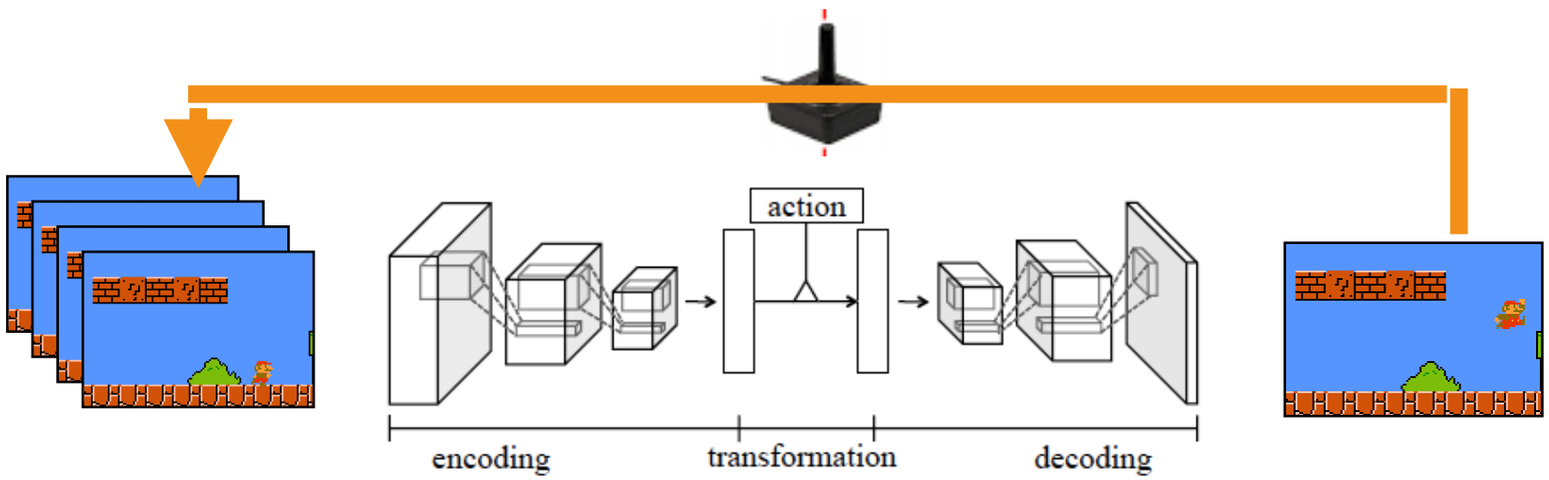
Minimizing error accumulation during unrolling



Unroll the model by feeding the prediction back as input!

Action-Conditional Video Prediction using Deep Networks in Atari Games, *Oh et al.*

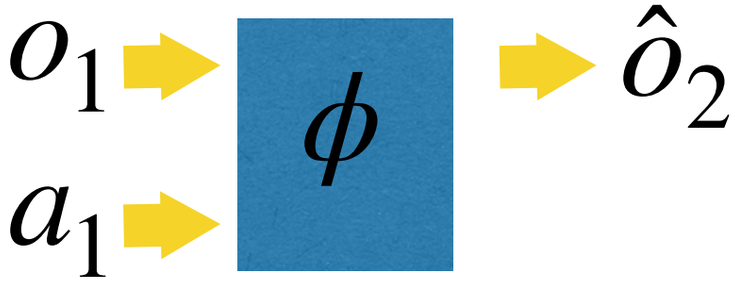
Minimizing error accumulation during unrolling



Q: Can I train my model using tuples (o, a, o') and at test time unroll it over time?
A: No, we will have distribution shift, same as in imitation learning: tiny mistakes will soon cause the model to drift

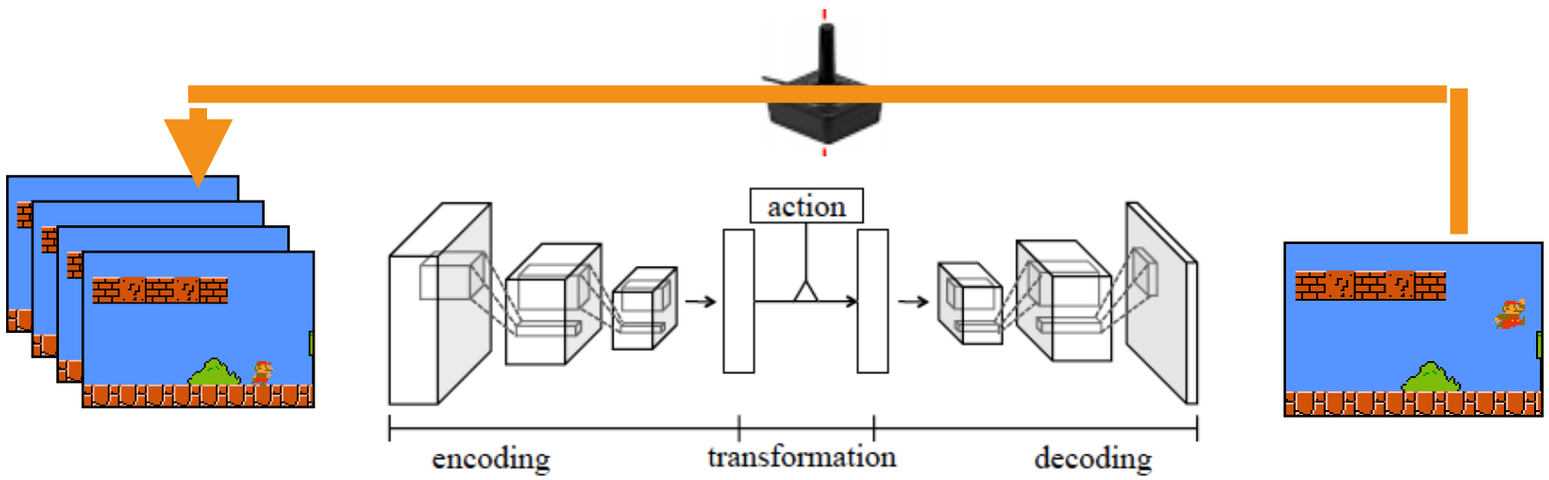
Solution: Progressively increase the unrolling horizon k at training time so that the model learns to handle its mistakes:

$$\mathscr{L}(\phi) = \frac{1}{N}\sum_{i=1}^{N} \|f(a_1^i, o_1^i; \phi) - o_2^i\|$$



Action-Conditional Video Prediction using Deep Networks in Atari Games, *Oh et al.*

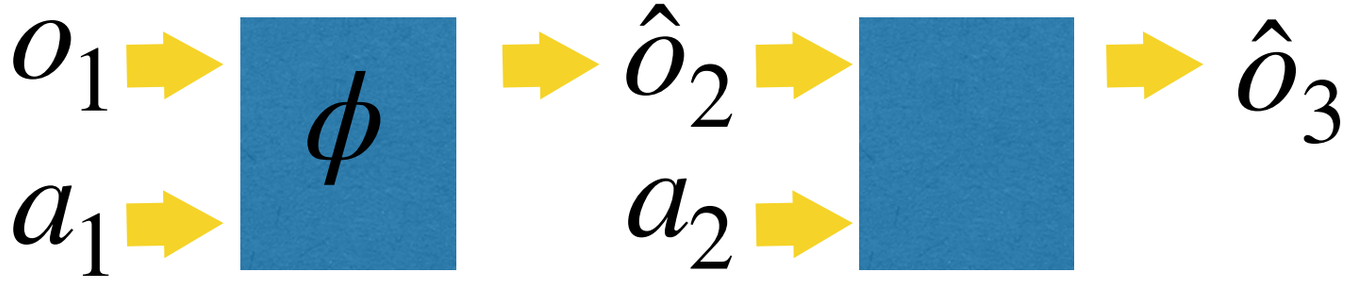
How to train our model so that unrolling works



Q: Can I train my model using tuples (o, a, o') and at test time unroll it over time?
A: No, we will have distribution shift, same as in imitation learning: tiny mistakes will soon cause the model to drift

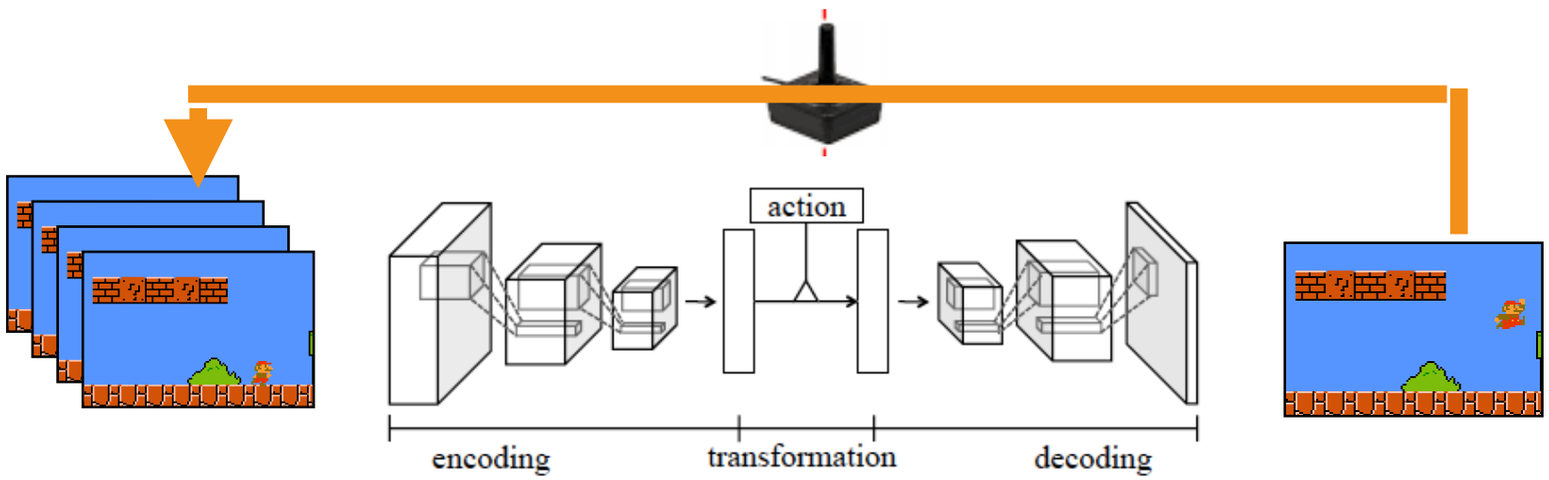
Solution: Progressively increase the unrolling horizon k at training time so that the model learns to handle its mistakes:

$$\mathscr{L}(\phi) = \frac{1}{N}\sum_{i=1}^{N} \|f(a_2^i, f(a_1^i, o_1^i; \phi); \phi) - o_3^i\| + \|f(a_1^i, o_1^i; \phi) - o_2^i\|$$



Action-Conditional Video Prediction using Deep Networks in Atari Games, *Oh et al.*

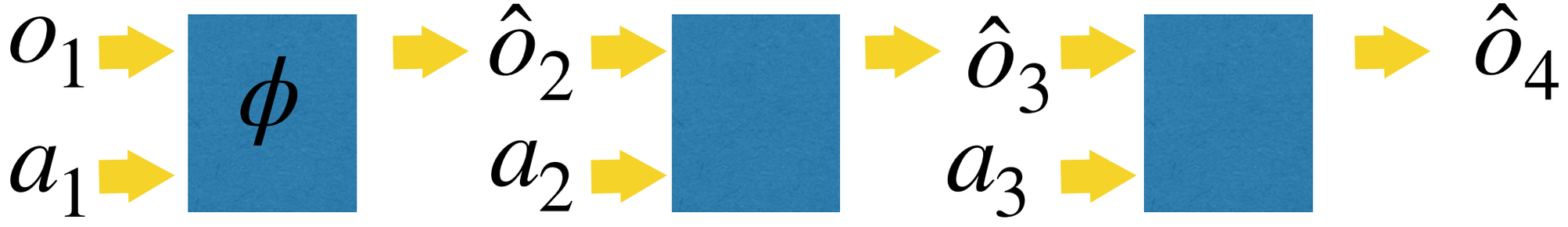
How to train our model so that unrolling works



Q: Can I train my model using tuples (o, a, o') and at test time unroll it over time?
A: No, we will have distribution shift, same as in imitation learning: tiny mistakes will soon cause the model to drift

Solution: Progressively increase the unrolling horizon k at training time so that the model learns to handle its mistakes:

$$\mathscr{L}(\phi) = \frac{1}{N}\sum_{i=1}^{N} \|f(a_3^i, f(a_2^i, f(a_1^i, o_1^i; \phi); \phi); \phi) - o_4^i\| + \|f(a_2^i, f(a_1^i, o_1^i; \phi); \phi) - o_3^i\| + \|f(a_1^i, o_1^i; \phi) - o_2^i\|$$



Action-Conditional Video Prediction using Deep Networks in Atari Games, *Oh et al.*

Small objects are missed, e.g., the bullets.

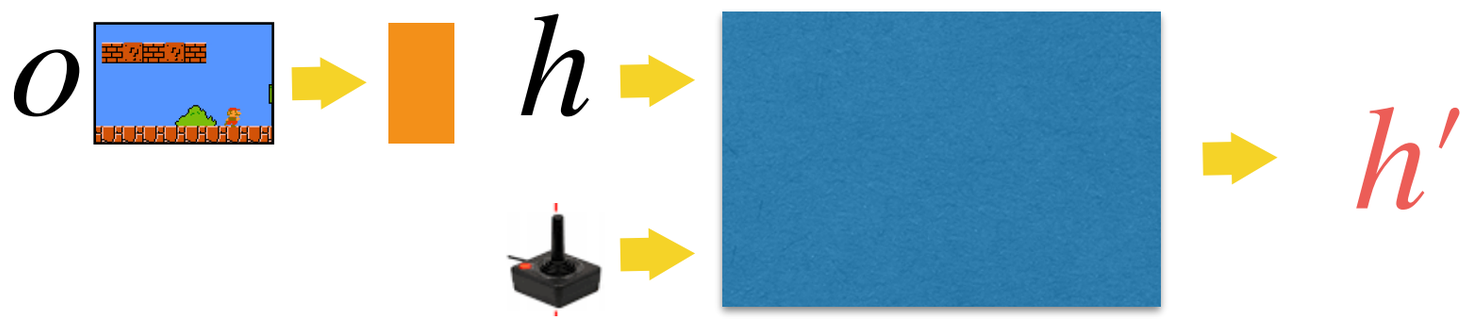
Q: Why?

A:They induce a tiny mean pixel prediction loss (despite the fact they may be very task-relevant!)

How can we get the model error to predict accurately the part of the observation relevant for the task and neglect irrelevant details?

Model Learning - 3 Qs always in mind

- What shall we be predicting?



Answers:

1.Future images (WXHX3 matrices)

2.Rewards

3.Object motions (in 2D)

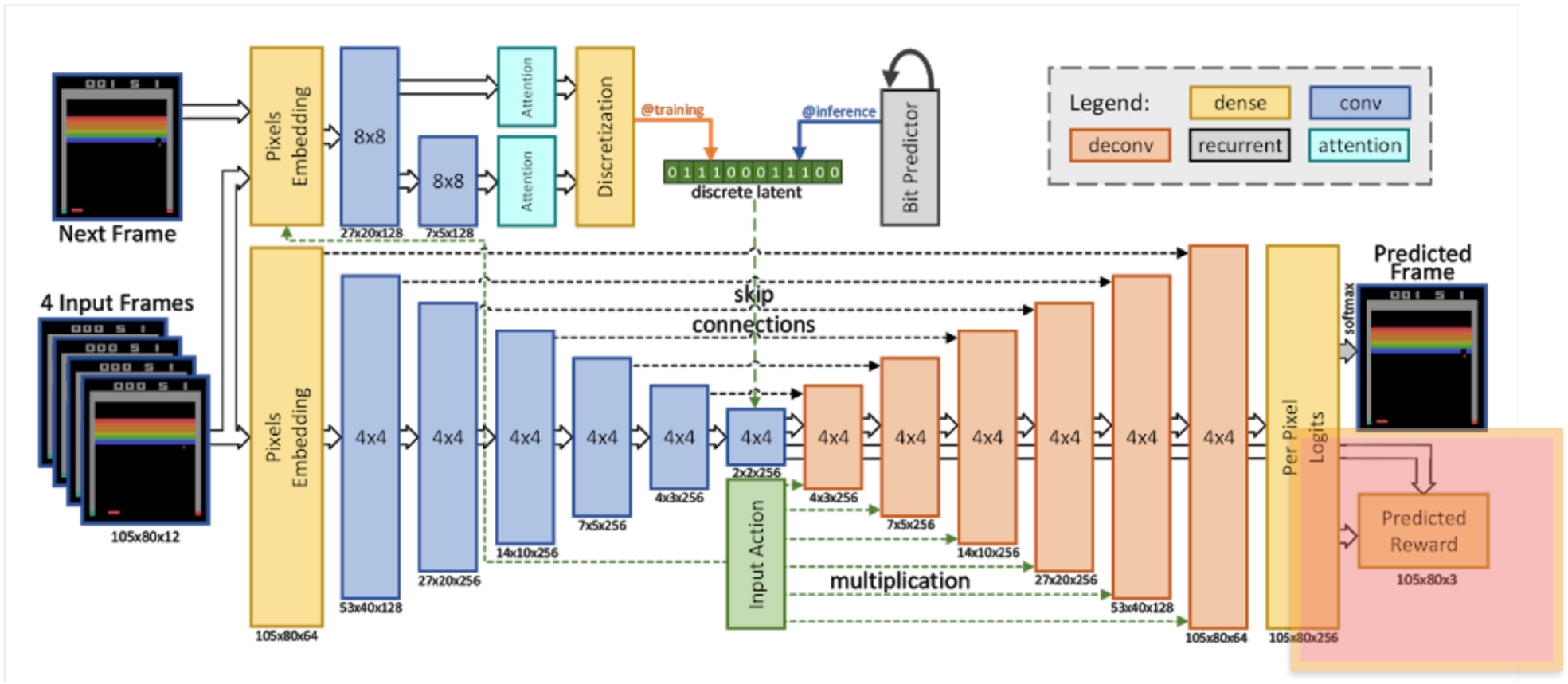
4.Image abstractions

5.Object abstractions

Model-Based Reinforcement Learning for Atari

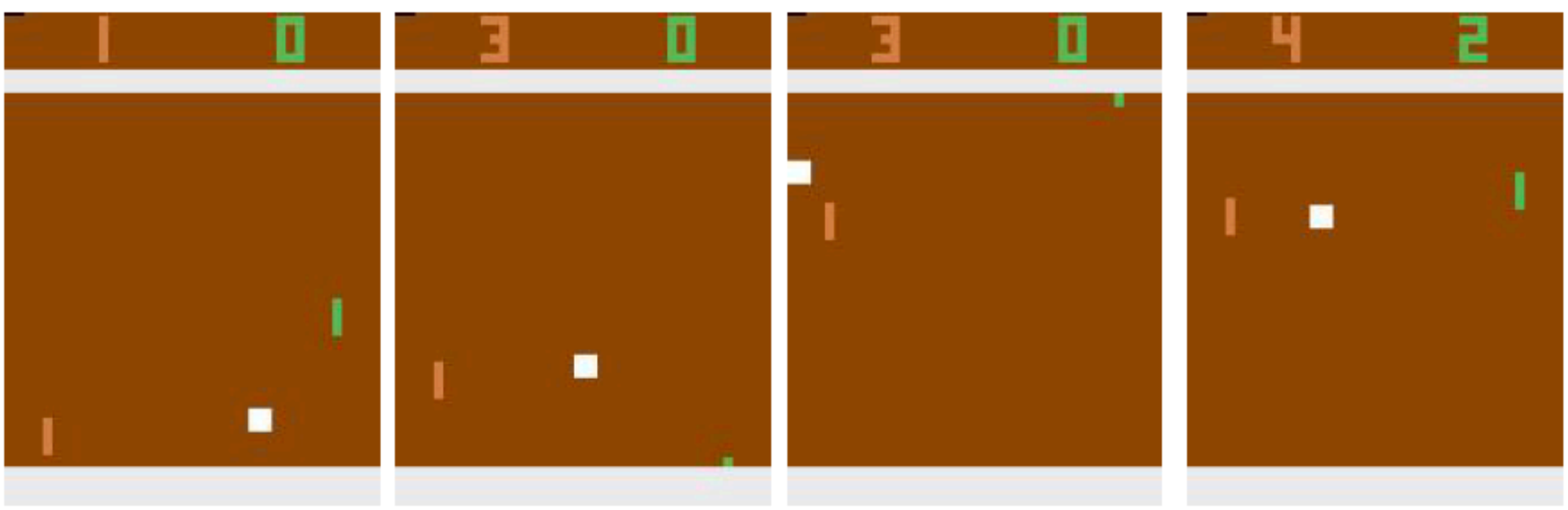
Łukasz Kaiser **Ryan Sepassi** Google Brain
Henryk Michalewski **Piotr Miłoś** University of Warsaw
Błażej Osiński deepsense.ai

Similar architecture as before but..



Reward-aware model learning loss!

- We train the dynamics model to generate a future sequence so that the rewards obtained from the simulated sequence agree with the rewards obtained in the ``real" (videogame) world. I put L2 loss on the rewards as opposed to just on pixels. This encourages to focus on objects that are too small and incur a tiny L2 pixel loss, but may be important for the game.
- (Nonetheless, they made the ball larger :-( )



results

Results

- Number of frames required to reach human performance

	PPO	Model-based
Breakout	800K	120K
Pong	1000K	500K
Freeway	200K	10K

results

LEARNING TO ACT BY PREDICTING THE FUTURE

Alexey Dosovitskiy
Intel Labs

Vladlen Koltun
Intel Labs

Main idea:

- You are provided with a set of measurements m (reward related) paired with input visual (and other sensory) observations.
- Measurements can be health, ammunition levels, enemies killed.
- Your goal can be expressed as a combination of those measurements.

measurement offsets for the next τ_n frames are the prediction targets: $\mathbf{f} = (\mathbf{m}_{t+\tau_1} - \mathbf{m}_t, \cdots, \mathbf{m}_{t+\tau_n} - \mathbf{m}_t)$

(multi) goal representation: $u(\mathbf{f}, \mathbf{g}) = \mathbf{g}^\top \mathbf{f}$

LEARNING TO ACT BY PREDICTING THE FUTURE

Alexey Dosovitskiy
Intel Labs

Vladlen Koltun
Intel Labs

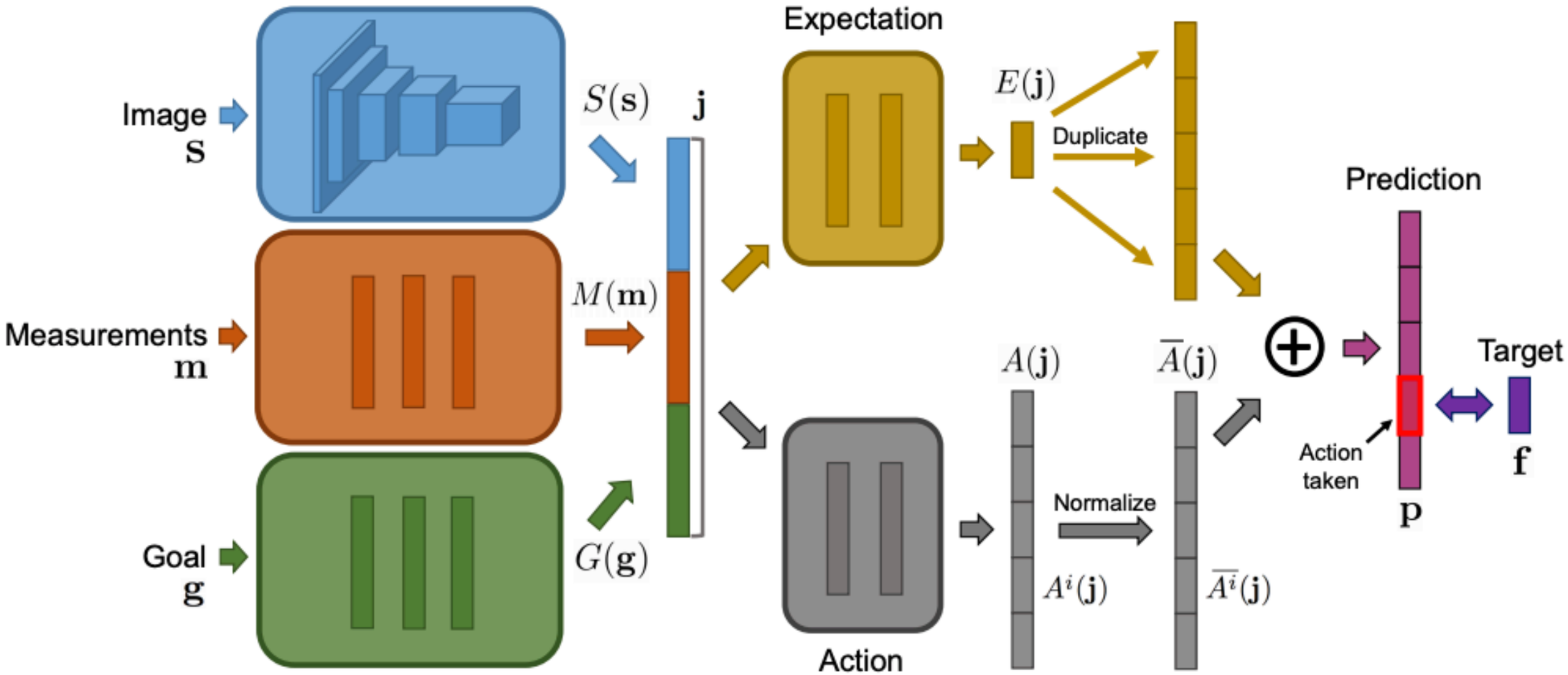
Train a deep predictor. No unrolling! One shot prediction of future values:

$$\mathscr{L}(\boldsymbol{\theta}) = \sum_{i=1}^{N} \left\| F\left(\mathbf{o}_i, a_i, \mathbf{g}_i; \boldsymbol{\theta}\right) - \mathbf{f}_i \right\|^2$$

No policy, direct action selection:

$$a_t = \arg\max_{a \in \mathscr{A}} \mathbf{g}^\top F\left(\mathbf{o}_t, a, \mathbf{g}; \boldsymbol{\theta}\right)$$

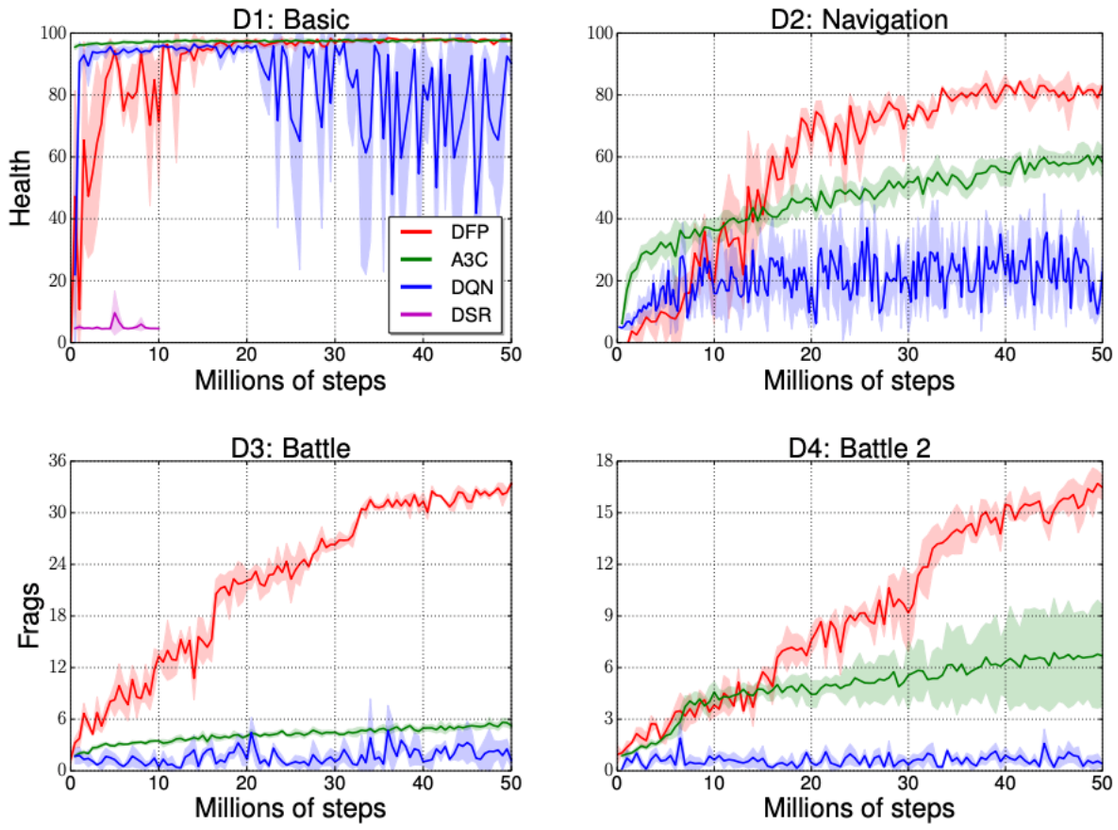
Learning dynamics of goal-related measurements



Action selection: $a_t = \arg\max_{a \in \mathcal{A}} \mathbf{g}^\top F\left(\mathbf{o}_t, a, \mathbf{g}; \boldsymbol{\theta}\right)$

Training: we learn the model using \epsilon-greedy exploration policy over the current best chosen actions.

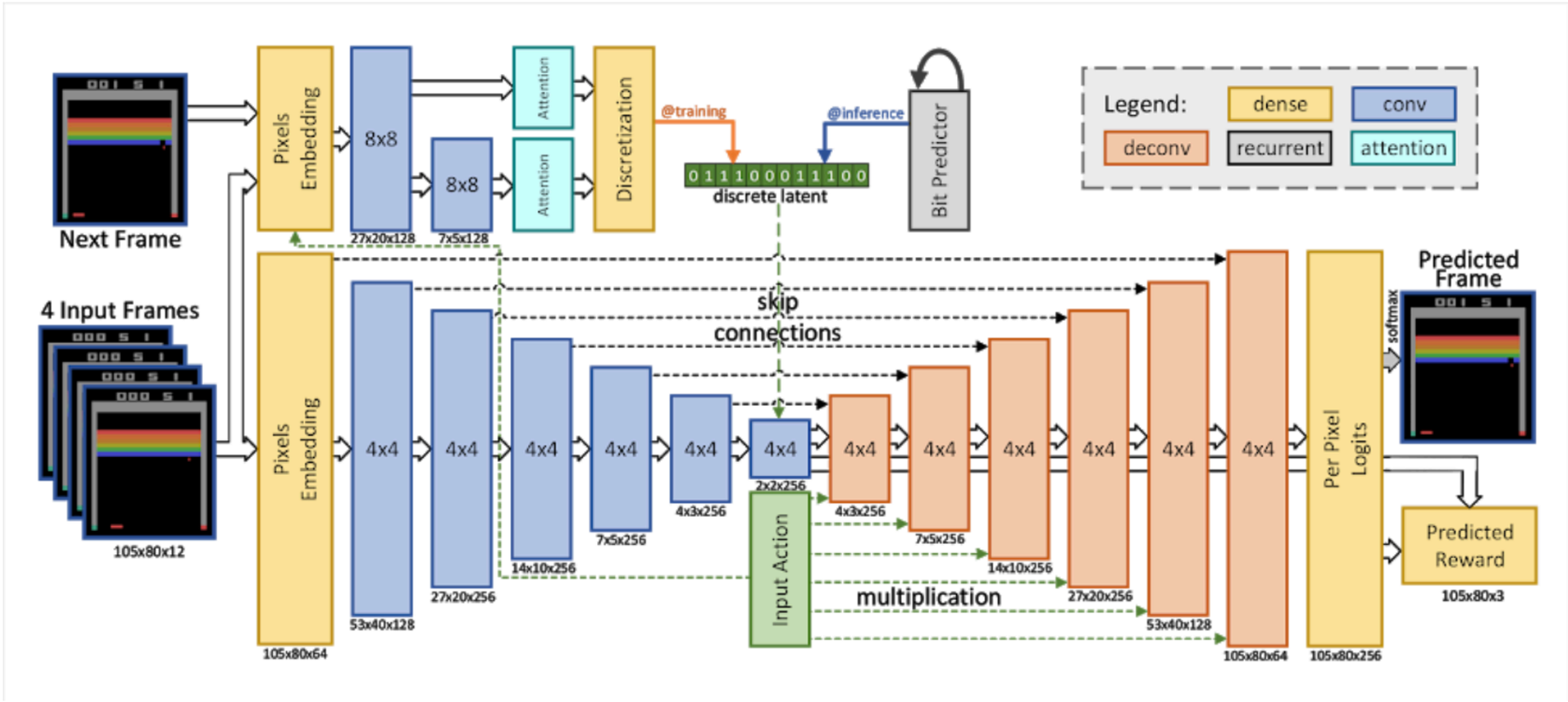
Learning dynamics of goal-related measurements



	D1 (health)	D2 (health)	D3 (frags)	D4 (frags)	steps/day
DQN	89.1 ± 6.4	25.4 ± 7.8	1.2 ± 0.8	0.4 ± 0.2	7M
A3C	$\mathbf{97.5 \pm 0.1}$	59.3 ± 2.0	5.6 ± 0.2	6.7 ± 2.9	80M
DSR	4.6 ± 0.1	–	–	–	1M
DFP	$\mathbf{97.7 \pm 0.4}$	$\mathbf{84.1 \pm 0.6}$	$\mathbf{33.5 \pm 0.4}$	$\mathbf{16.5 \pm 1.1}$	70M

Table 1: Comparison to prior work. We report average health at the end of an episode for scenarios D1 and D2, and average frags at the end of an episode for scenarios D3 and D4.

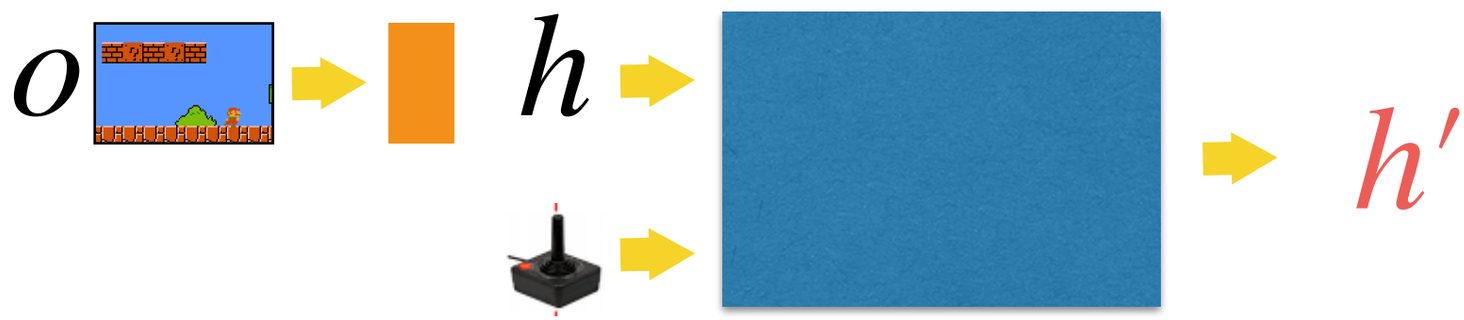
Generalization in convolutional frame prediction



- Can the model generalize to scenes with different number of objects?

Model Learning - 3 Qs always in mind

- What shall we be predicting?



Answers:

1.Future images (WXHX3 matrices)

2.Rewards

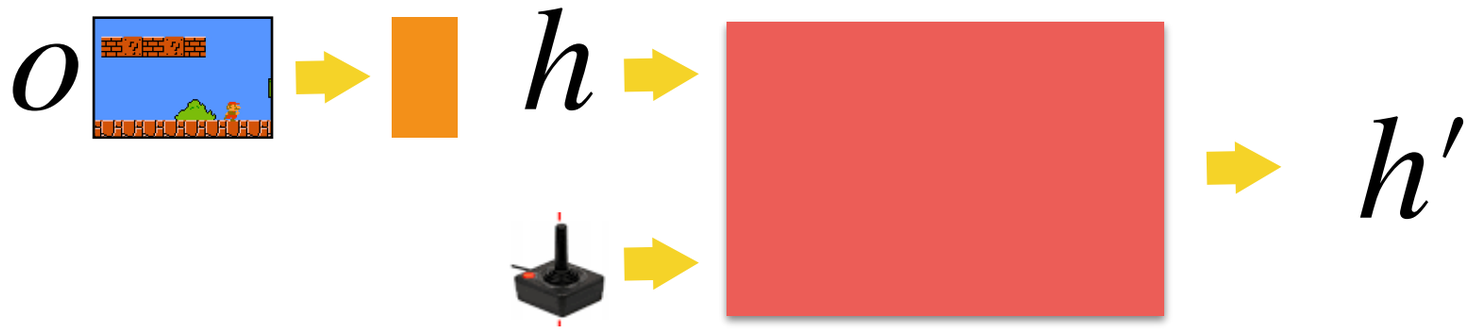
3.Object motions (in 2D)

4.Image abstractions

5.Object abstractions

Model Learning - 3 Qs always in mind

- What is the architecture of the model, what structural biases should we add to get it to generalize?



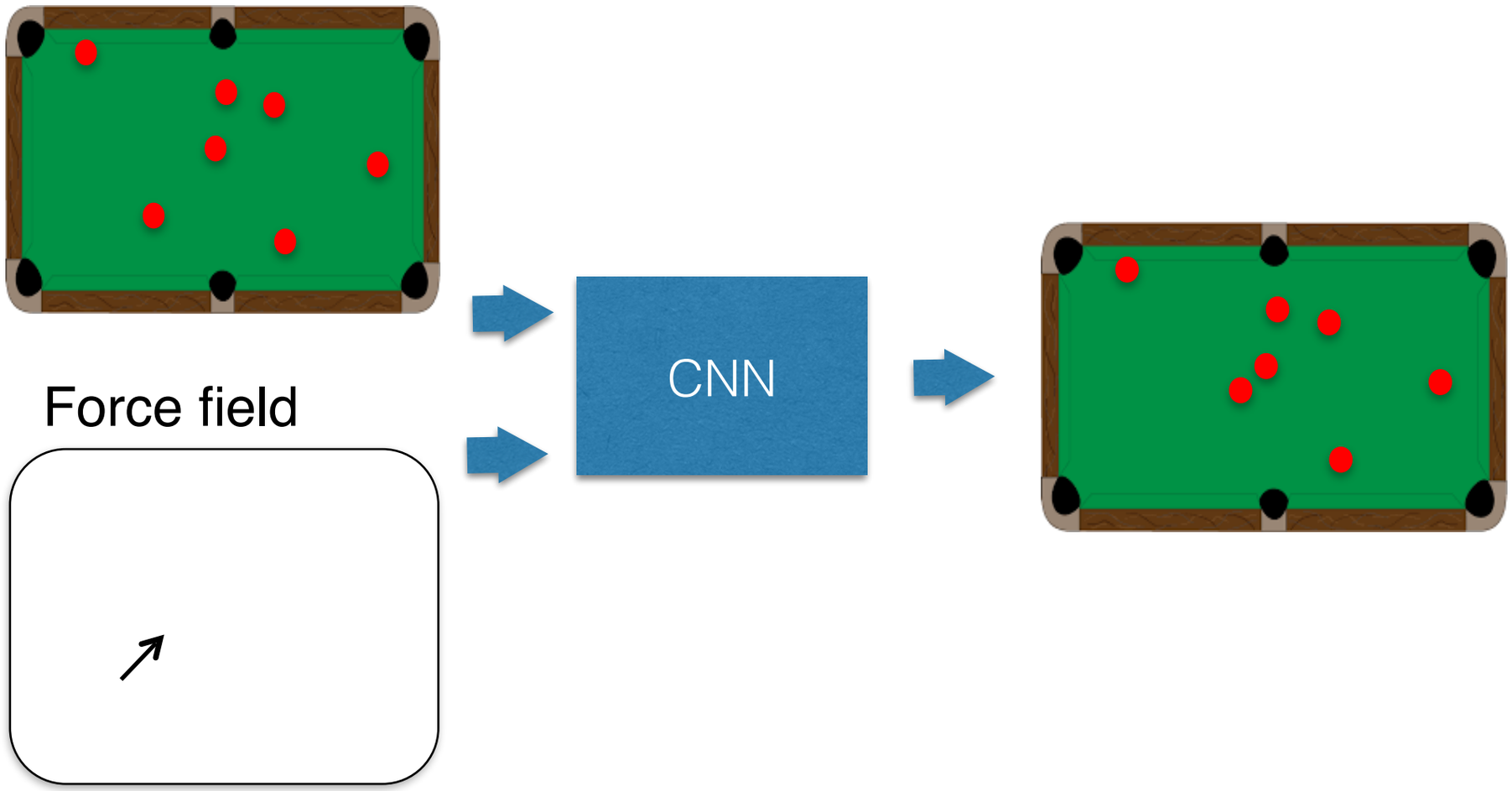
Answers:

1.CNNs

2.Object-centric CNNs

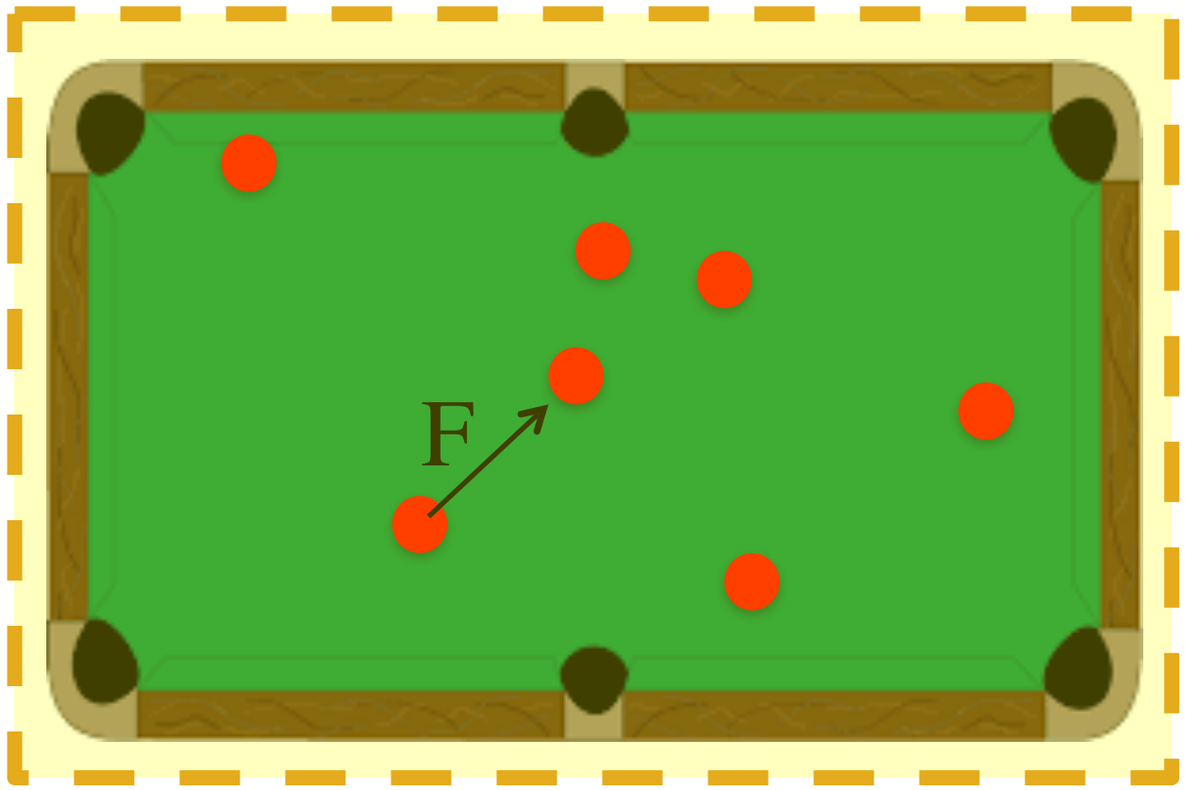
3.Graph neural nets

Problem with CNNs



Q: Will our model be able to generalize across different number of balls present?

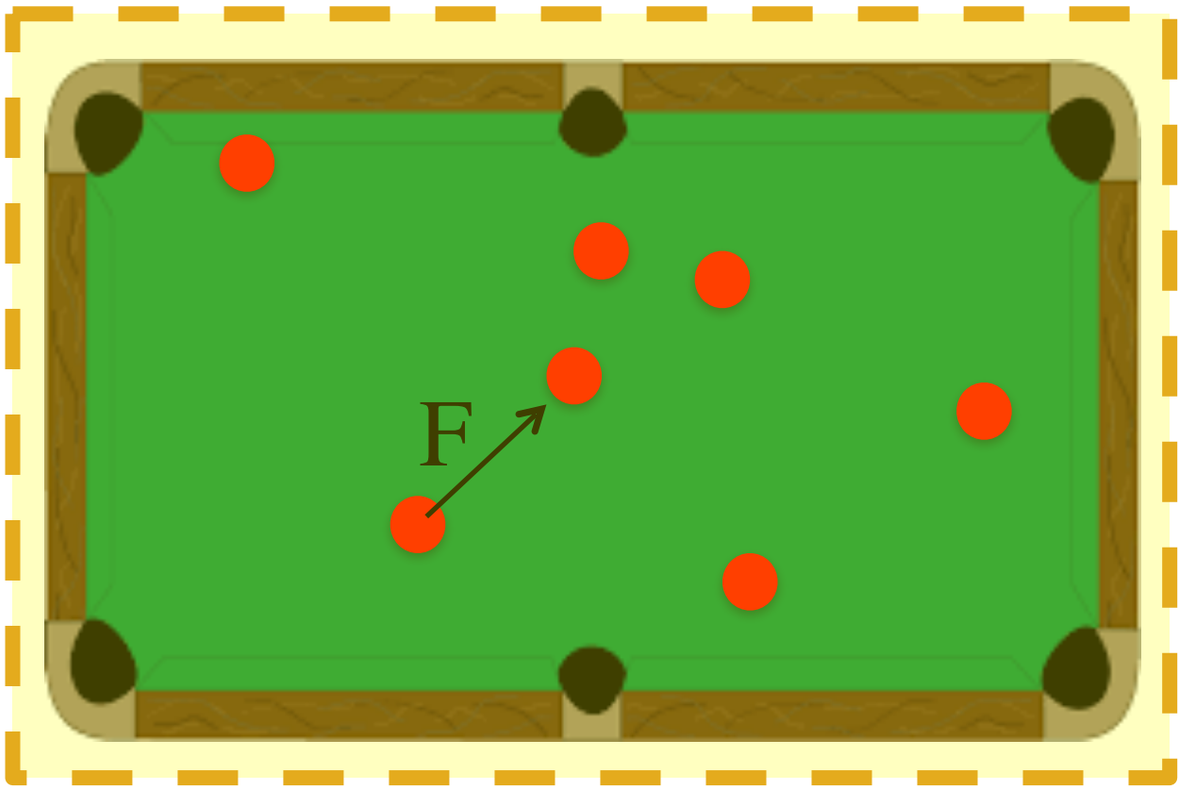
Frame-centric CNNs



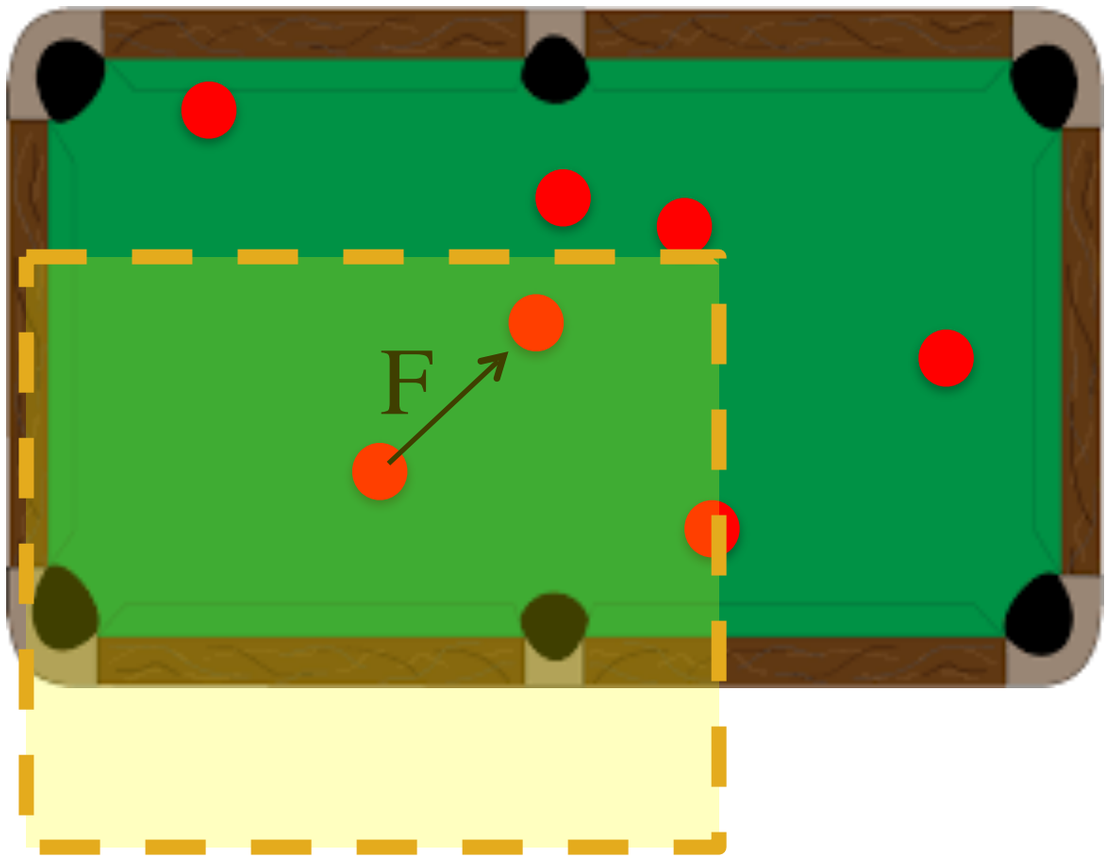
Frame-Centric Prediction

Q: Will our model be able to generalize across different number of balls present?

Object-centric CNNs



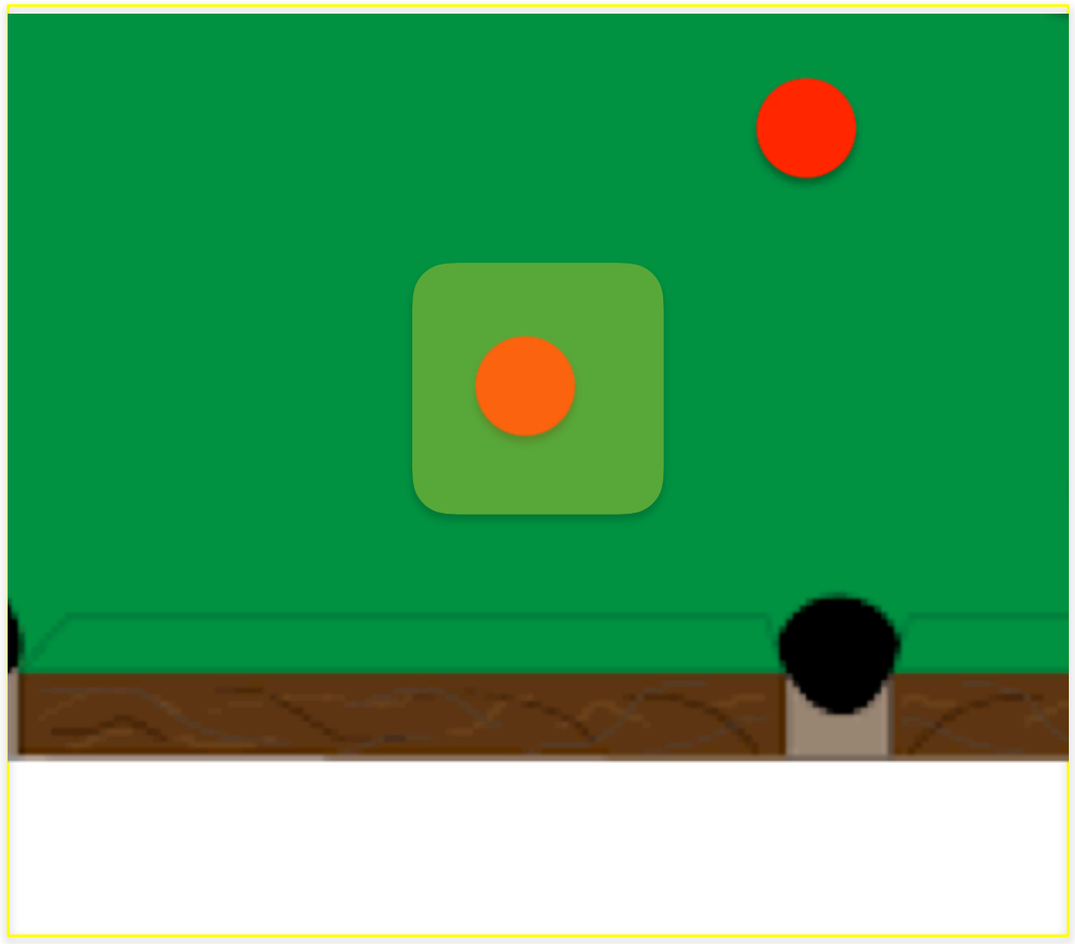
World-Centric Prediction



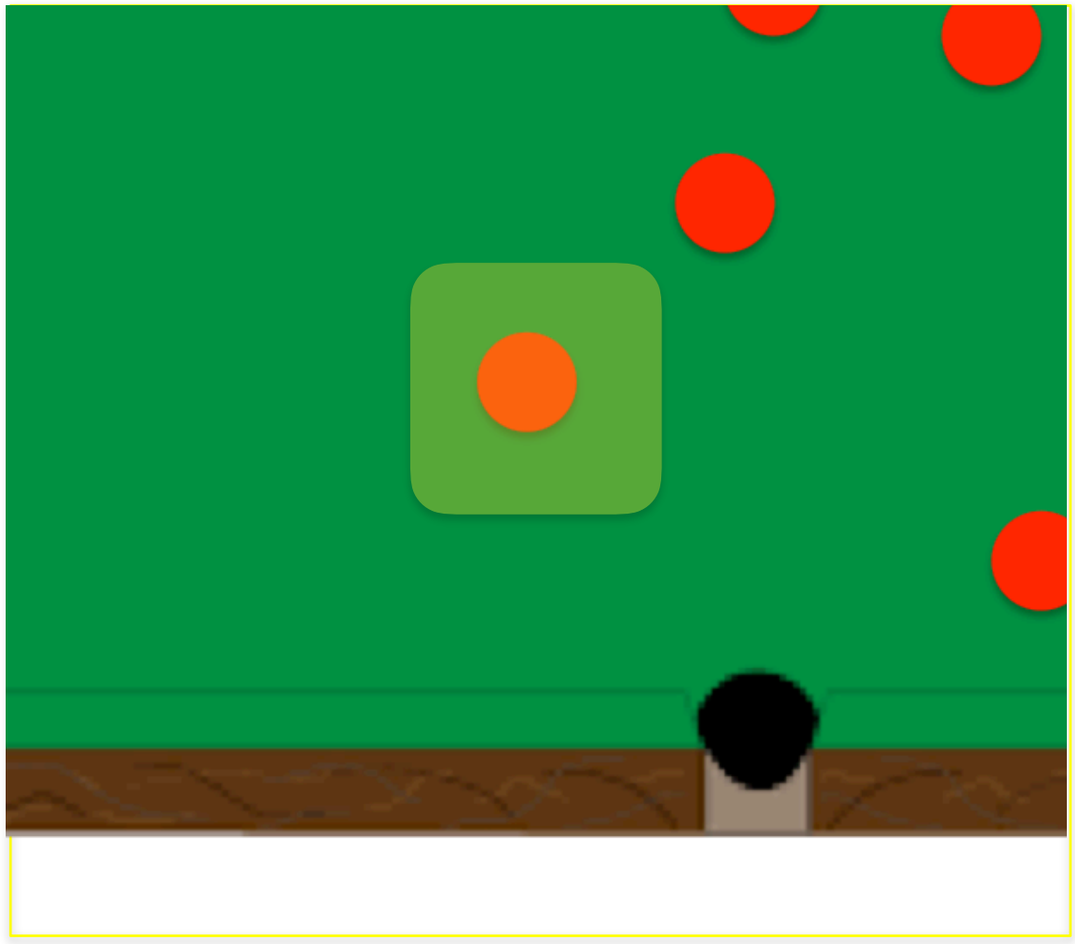
Object-Centric Prediction

The object-centric CNN will be applied to each object in the scene

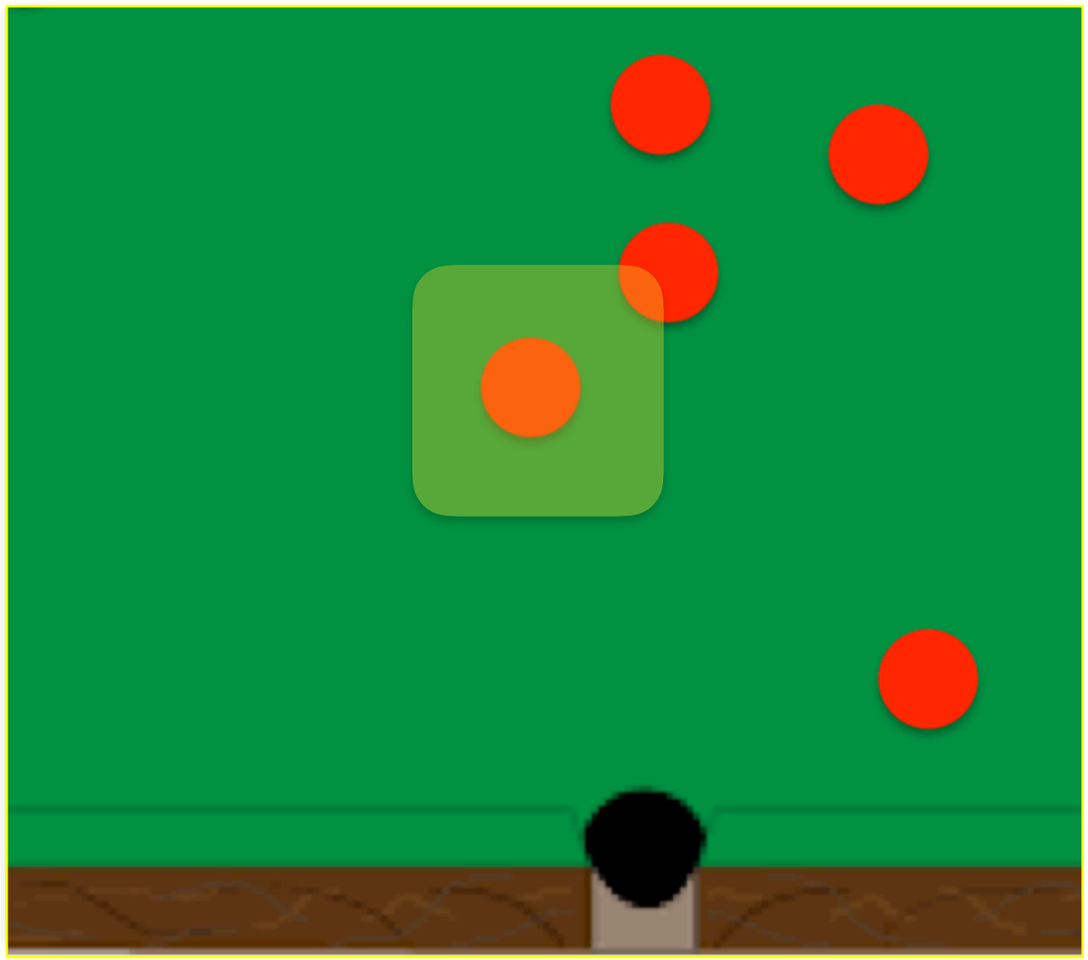
Context around the object is captured by using large windows around the object of interest



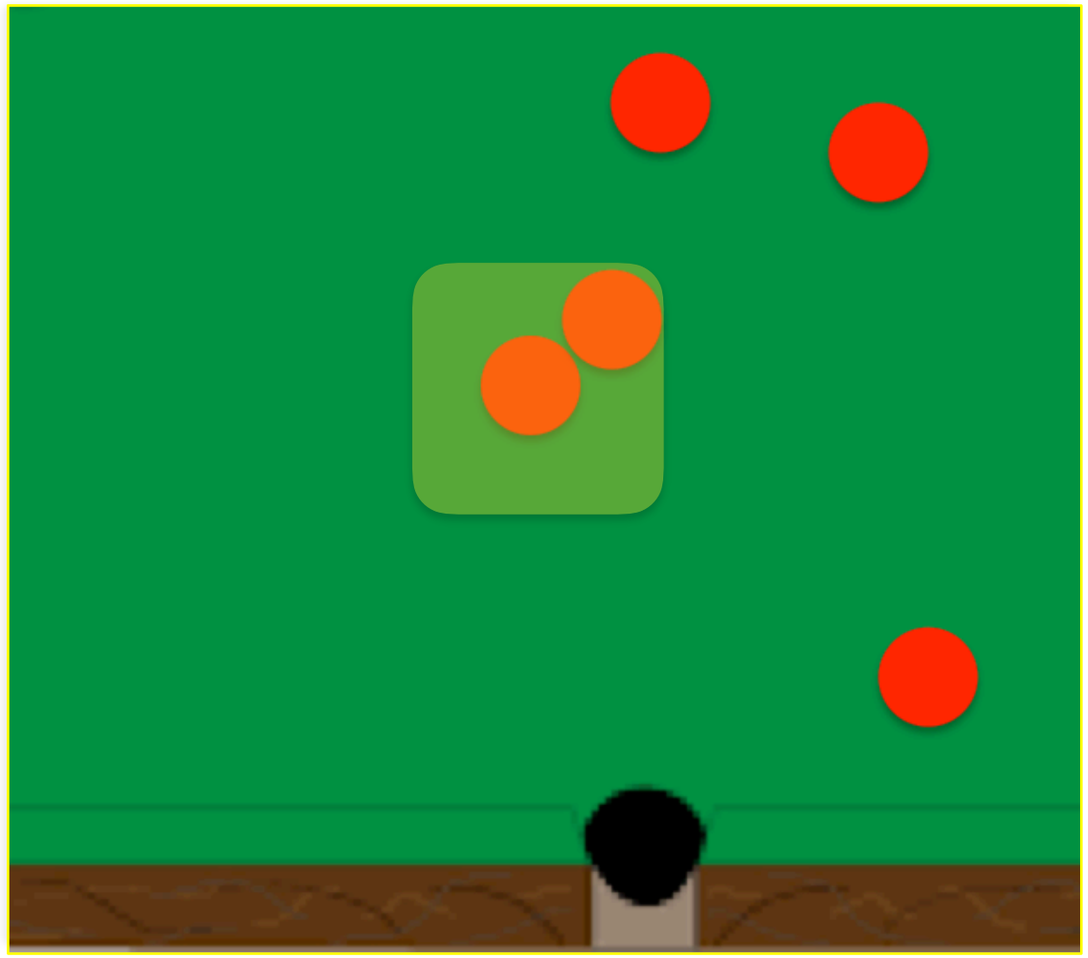
Context around the object is captured by using large windows around the object of interest



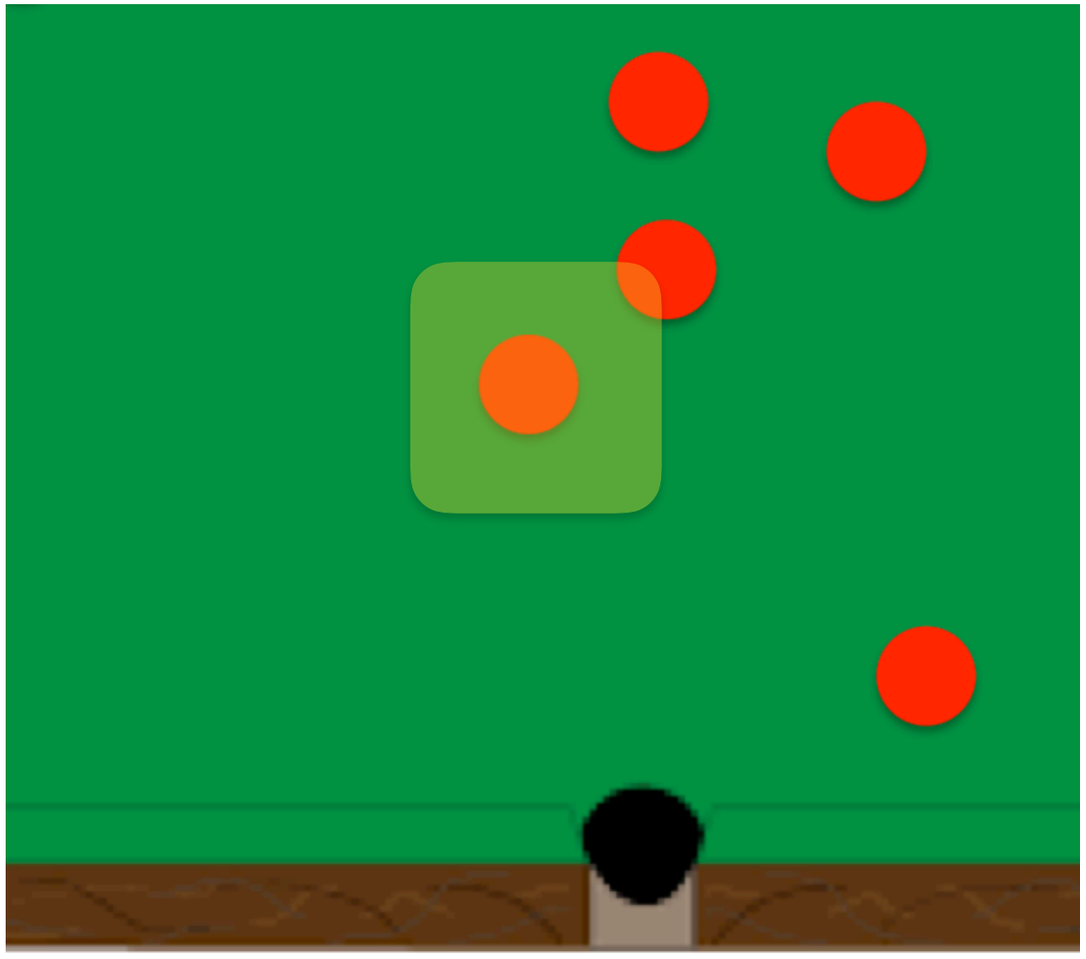
Context around the object is captured by using large windows around the object of interest



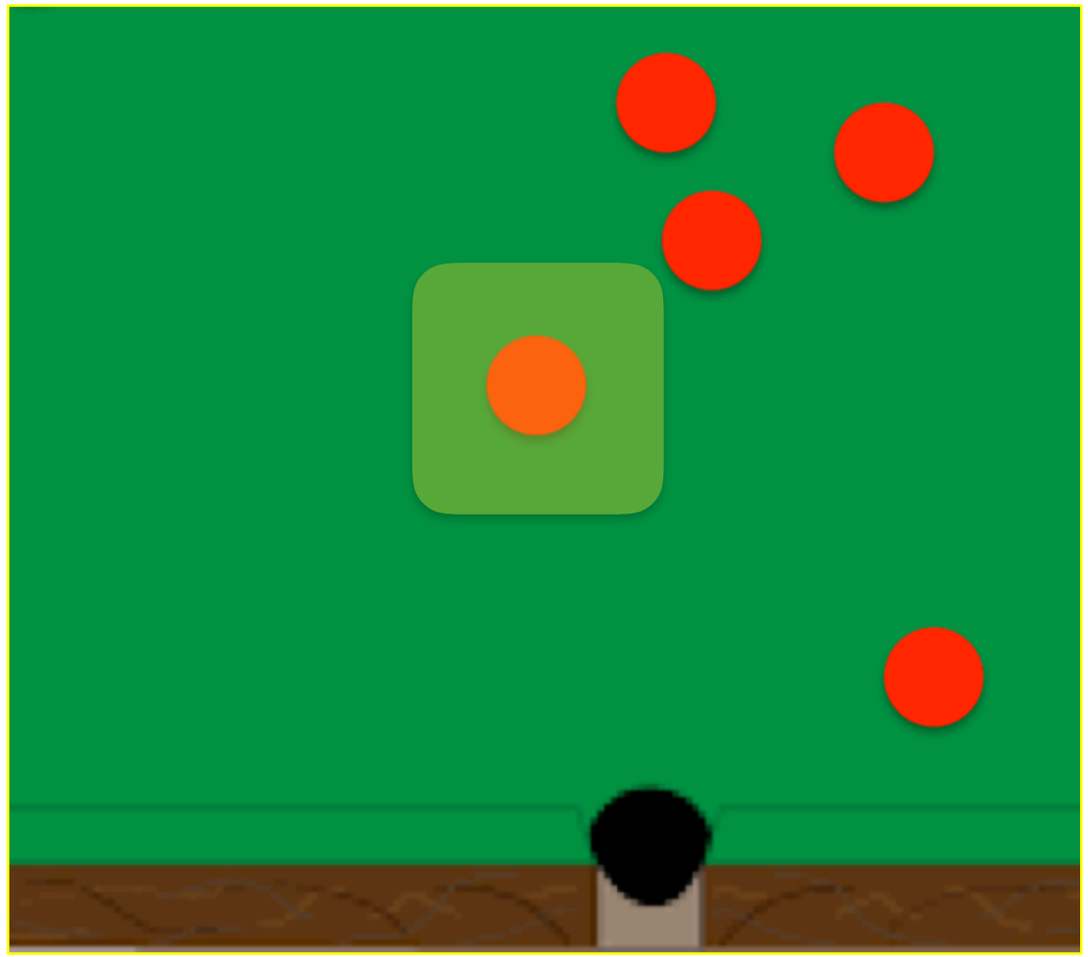
Context around the object is captured by using large windows around the object of interest



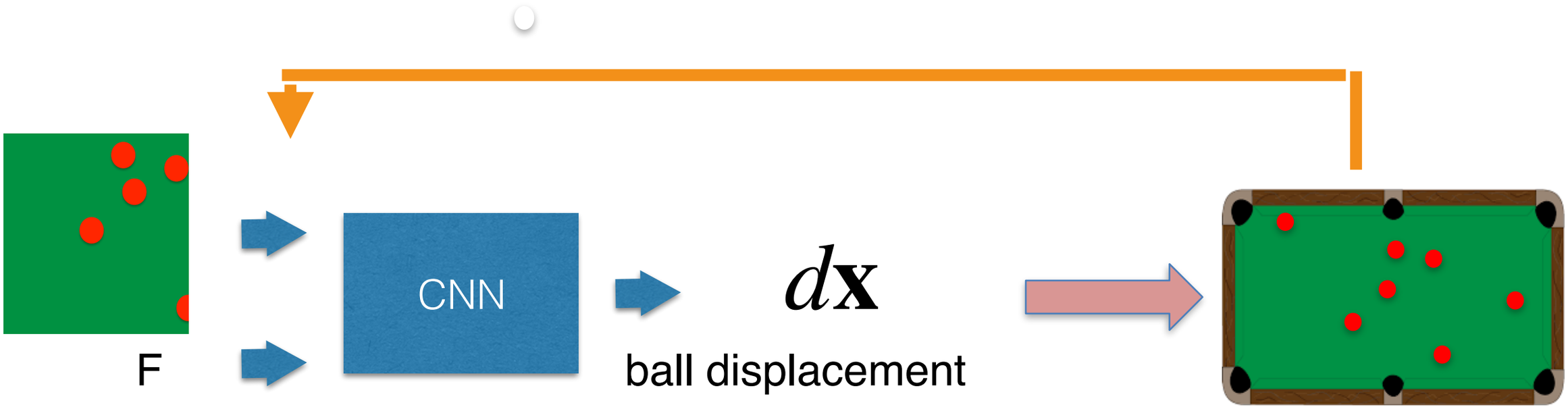
Context around the object is captured by using large windows around the object of interest



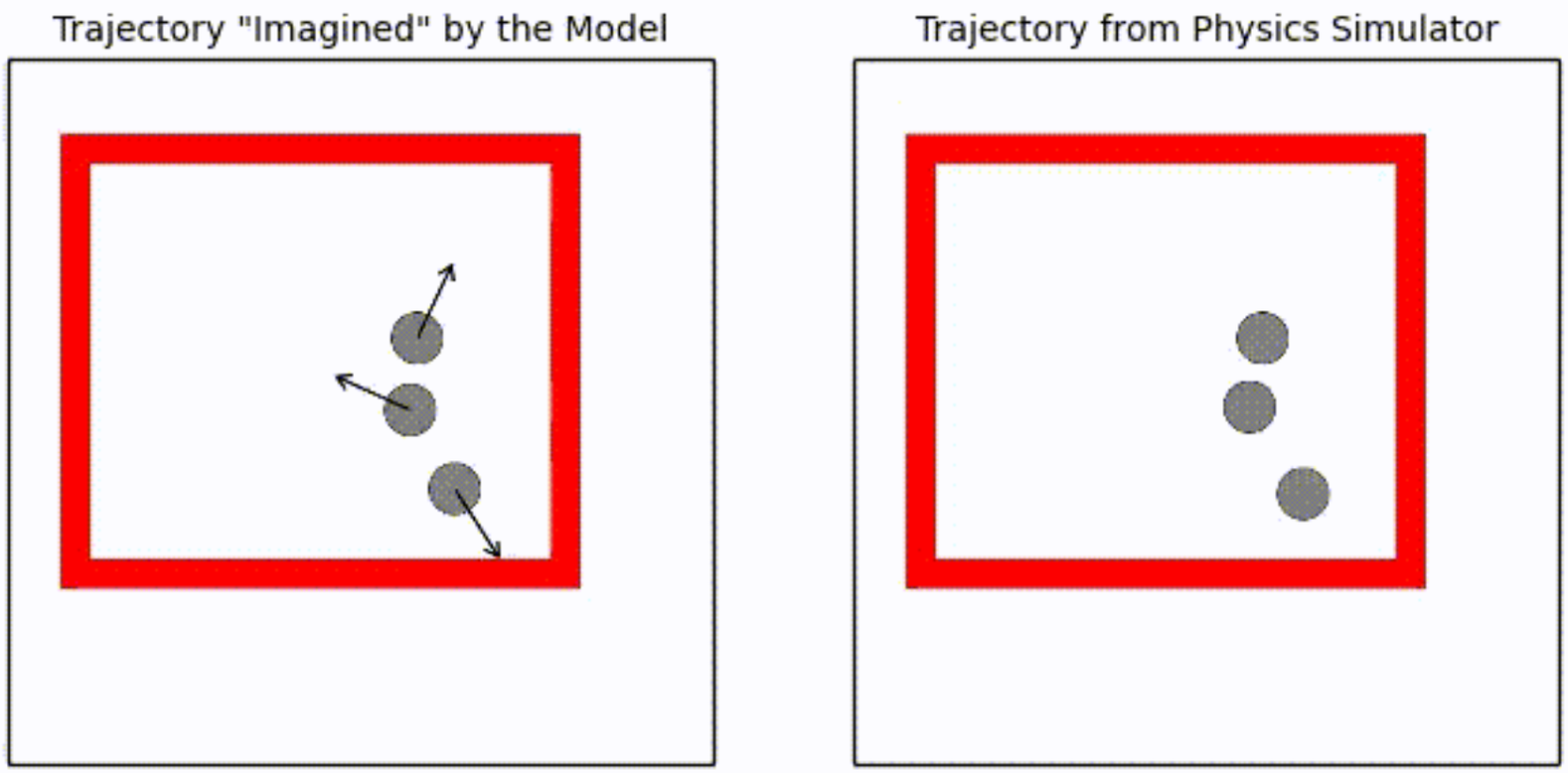
Context around the object is captured by using large windows around the object of interest



Unrolling with object-centric CNN

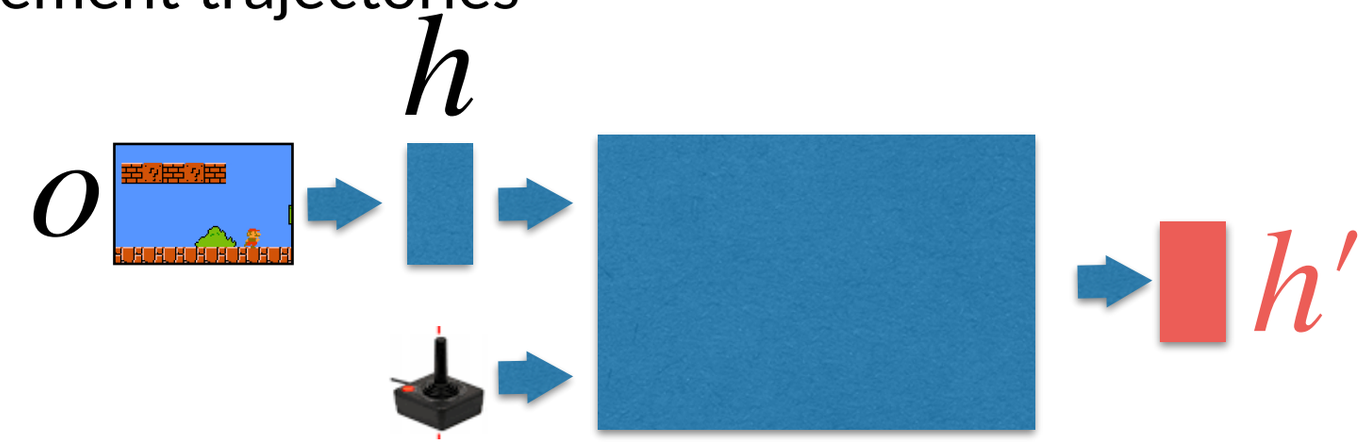


- The object-centric CNN is shared across all objects in the scene.
- We apply it one object at a time to predict the object's future displacement.
- We then copy paste the ball at the predicted location, and feed back as input.

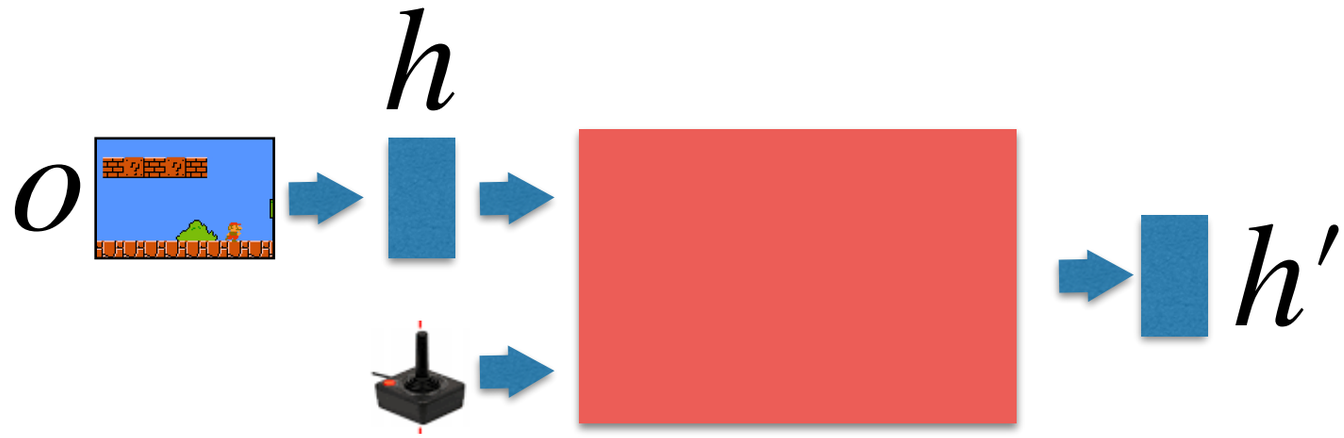

Trajectory "Imagined" by the Model
Trajectory from Physics Simulator

Object-centric Billiard Dynamics

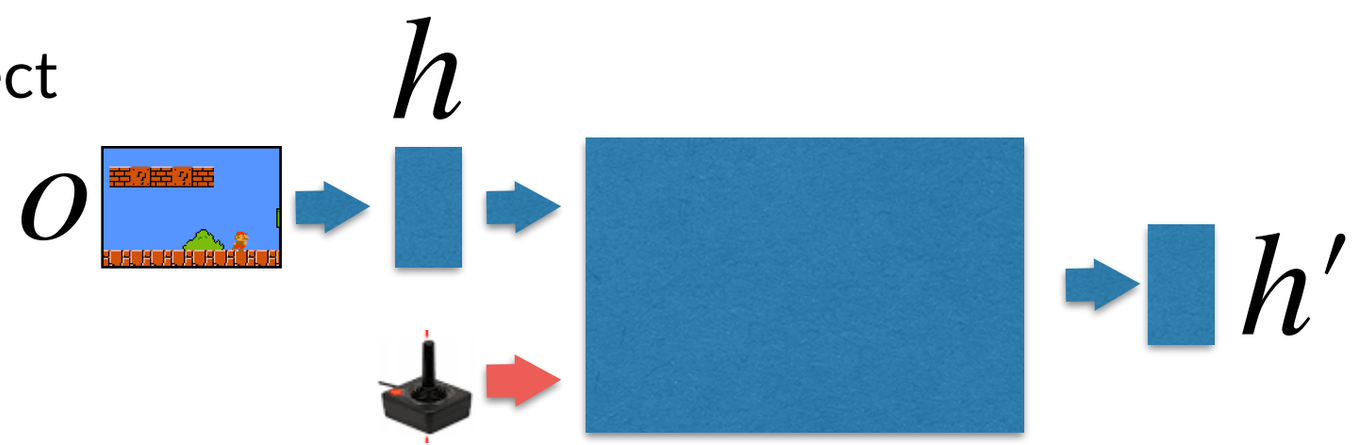
- We predicted object displacement trajectories

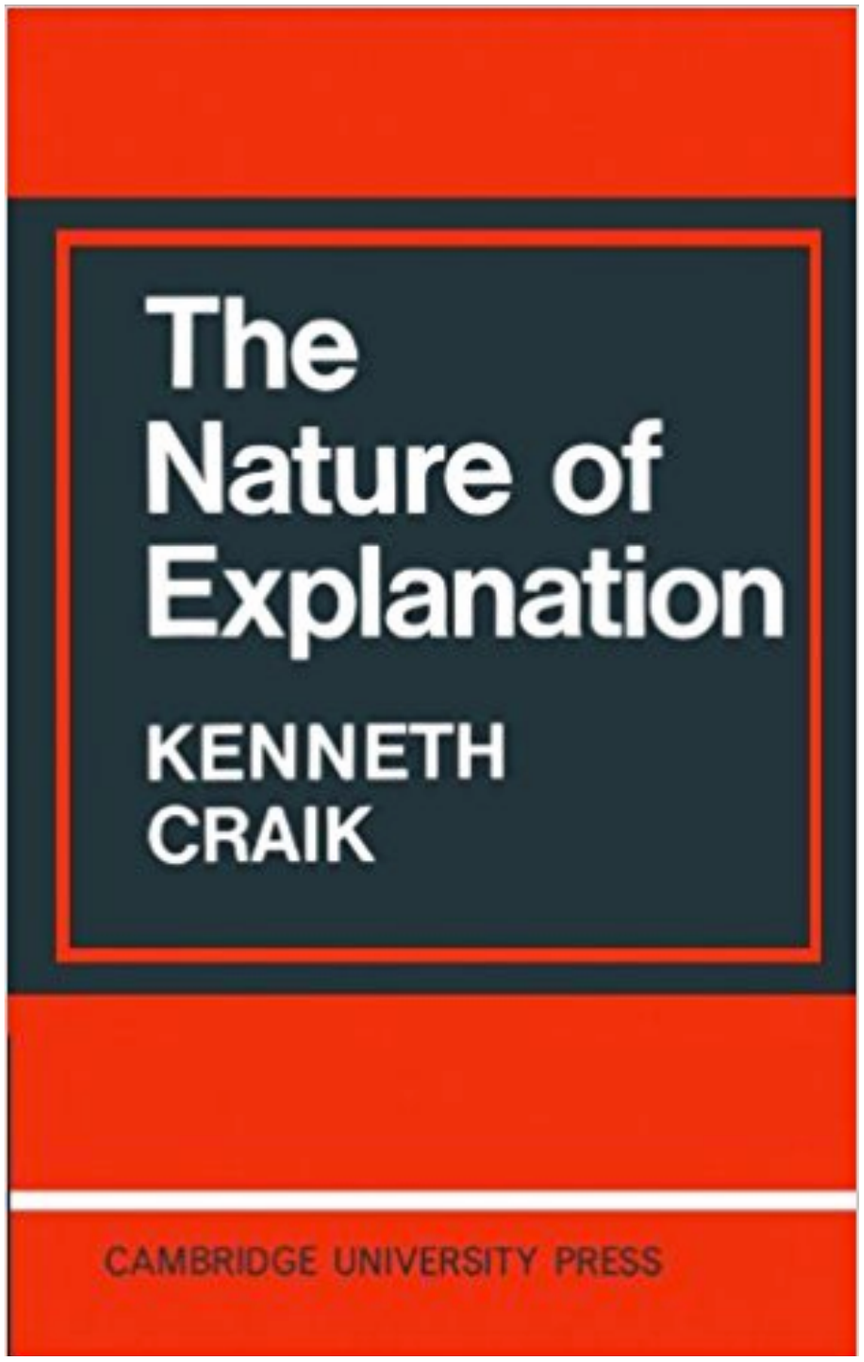


- We had one CNN per object in the scene, shared the weights across objects



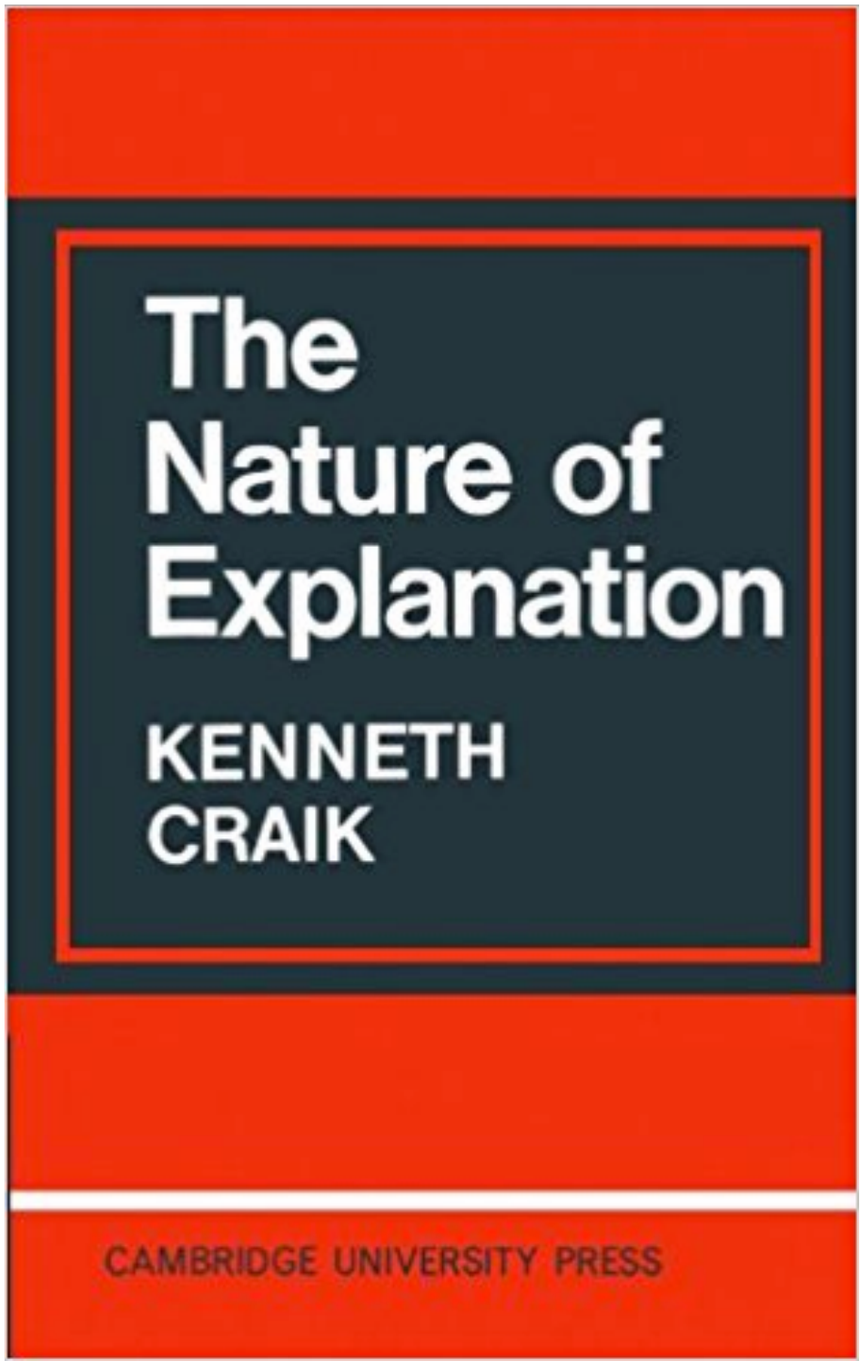
- A force applied to each object





Kenneth Craik, "The Nature of Explanation", 1943:

"If the organism carries a **'small-scale model' of external reality** and of **its own possible actions** within its head, it is able to **try out various alternatives, conclude which is the best of them, react to future situations before they arise,** utilize the knowledge of past events in dealing with the present and future, and in every way to react in a much fuller, safer, and more competent manner to the emergencies which face it." (pg 61)



Kenneth Craik, "The Nature of Explanation", 1943:

"If the organism carries a **'small-scale model' of external reality** and of **its own possible actions** within its head, it is able to **try out various alternatives**, **conclude which is the best of them**, **react to future situations before they arise**, utilize the knowledge of past events in dealing with the present and future, and in every way to react in a much fuller, safer, and more competent manner to the emergencies which face it." (pg 61)

"This concept of **'thinghood'** is of fundamental importance for any **theory of thought**." (pg 77)

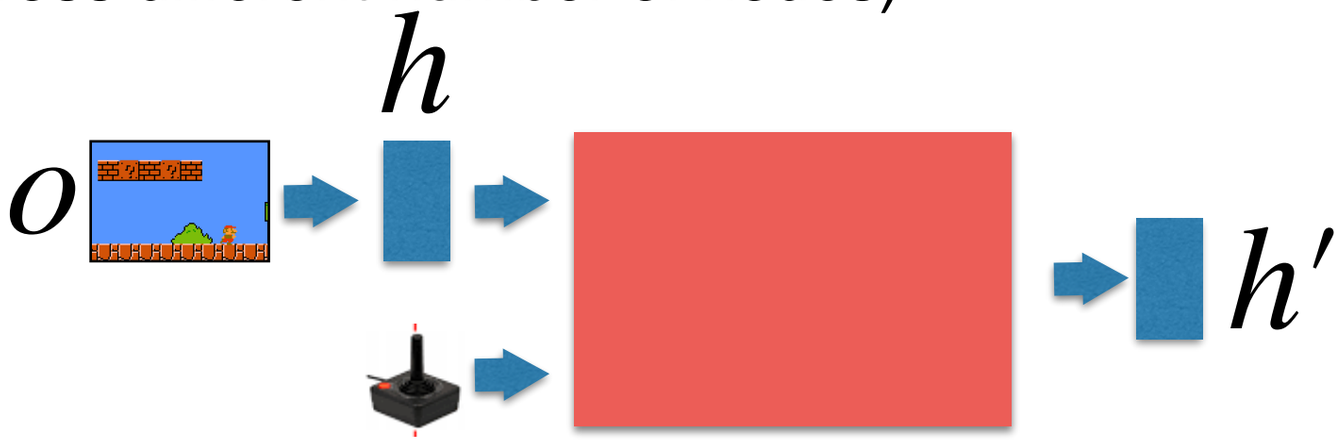
Cross-object interactions

How can we encode cross-object relations?

1. using large context windows around each object (this is what we just used)
2. using graph neural networks!

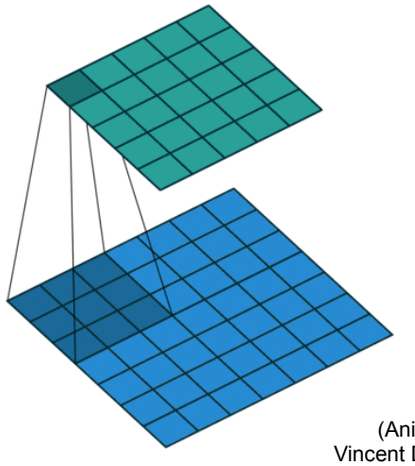
Scenes as graphs

- The node update function is shared across all object and robot joint nodes (thus we can generalize across different number of nodes)

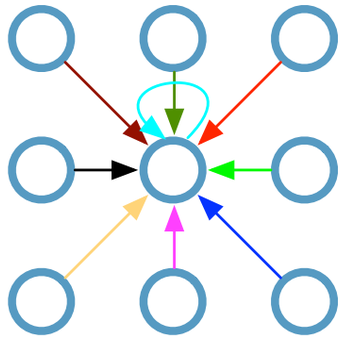


From CNNs to GNNs

Single CNN layer with 3x3 filter:

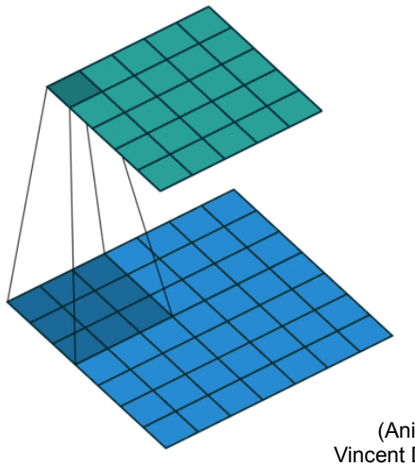


(Animation by Vincent Dumoulin)

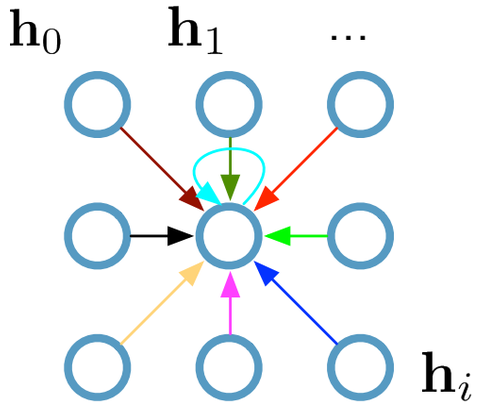


From CNNs to GNNs

Single CNN layer with 3x3 filter:

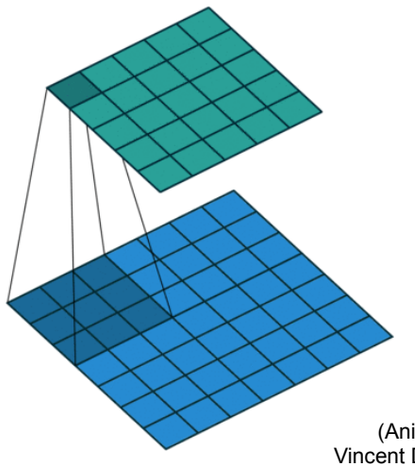


(Animation by Vincent Dumoulin)

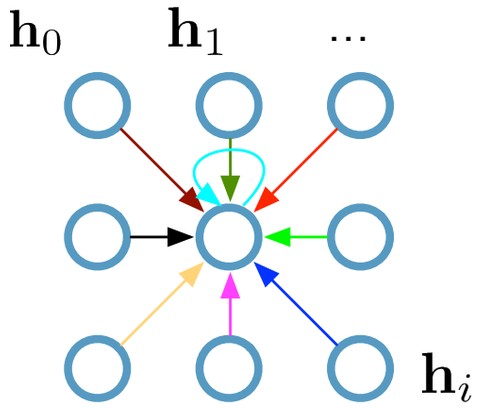


From CNNs to GNNs

Single CNN layer with 3x3 filter:



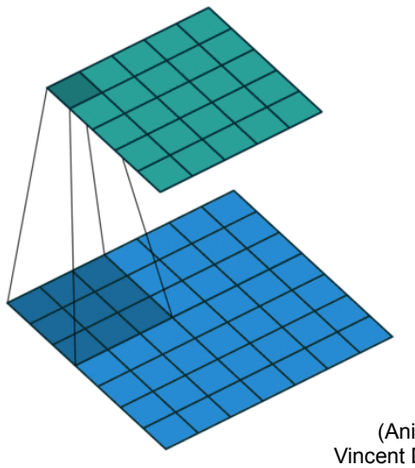
(Animation by Vincent Dumoulin)



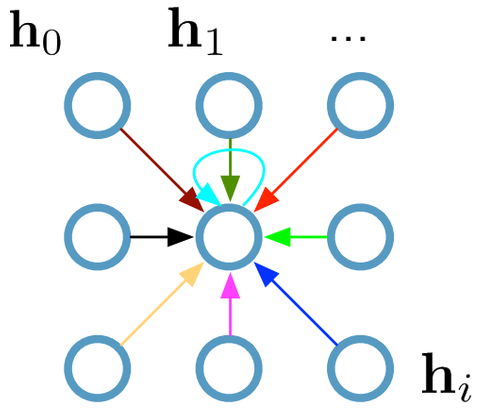
$\mathbf{h}_i \in \mathbb{R}^F$ are (hidden layer) activations of a pixel/node

From CNNs to GNNs

Single CNN layer with 3x3 filter:



(Animation by Vincent Dumoulin)



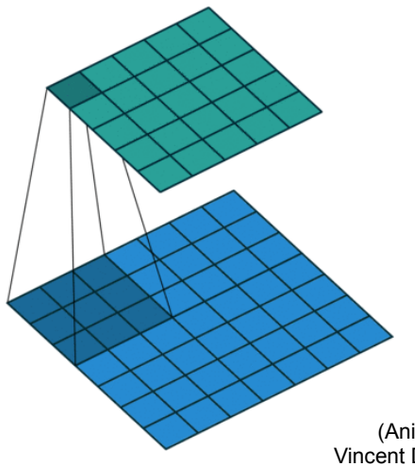
Update for a single pixel:

- Transform messages individually $\mathbf{W}_i\mathbf{h}_i$
- Add everything up $\sum_i \mathbf{W}_i\mathbf{h}_i$

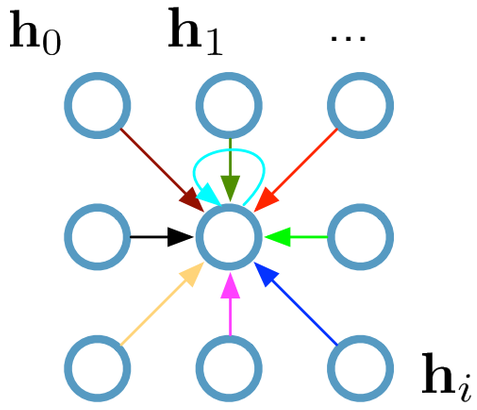
$\mathbf{h}_i \in \mathbb{R}^F$ are (hidden layer) activations of a pixel/node

From CNNs to GNNs

Single CNN layer with 3x3 filter:



(Animation by Vincent Dumoulin)



Update for a single pixel:

- Transform messages individually $\mathbf{W}_i\mathbf{h}_i$
- Add everything up $\sum_i \mathbf{W}_i\mathbf{h}_i$

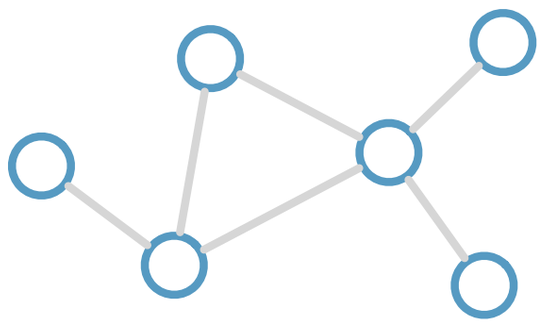
$\mathbf{h}_i \in \mathbb{R}^F$ are (hidden layer) activations of a pixel/node

Full update:

$$\mathbf{h}_4^{(l+1)} = \sigma\left(\mathbf{W}_0^{(l)}\mathbf{h}_0^{(l)} + \mathbf{W}_1^{(l)}\mathbf{h}_1^{(l)} + \cdots + \mathbf{W}_8^{(l)}\mathbf{h}_8^{(l)}\right)$$

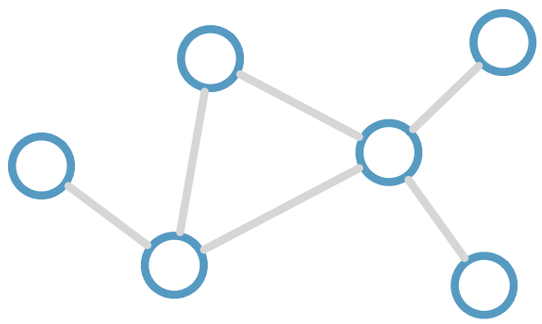
From CNNs to GNNs

Consider this
undirected graph:

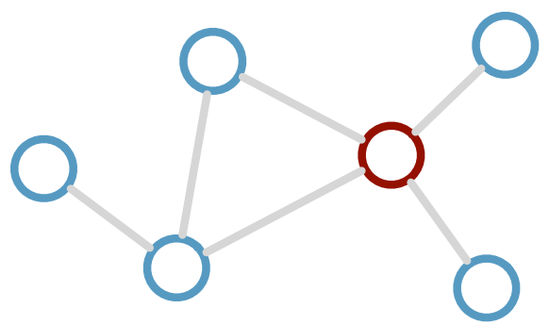


From CNNs to GNNs

Consider this undirected graph:

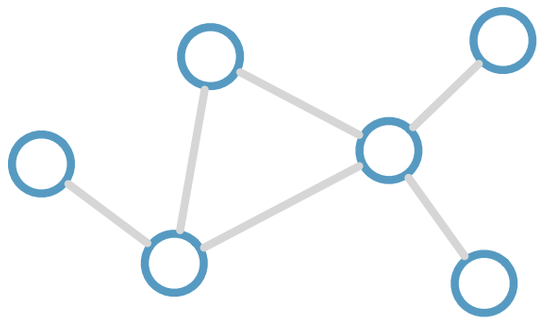


Calculate update for node in red:

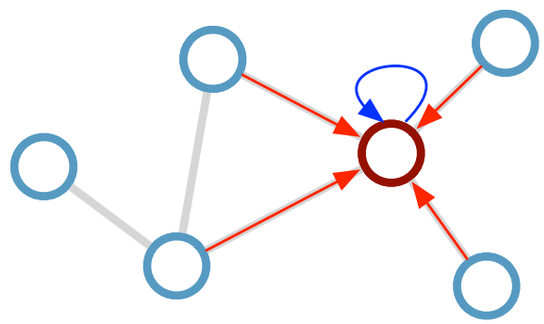


From CNNs to GNNs

Consider this undirected graph:

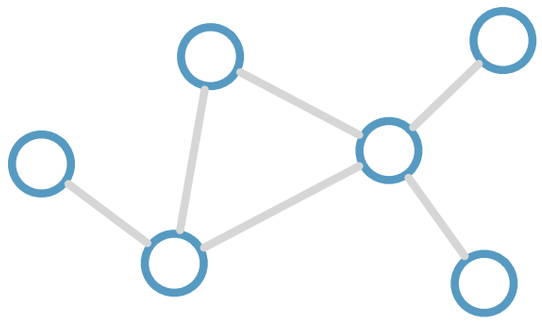


Calculate update for node in red:

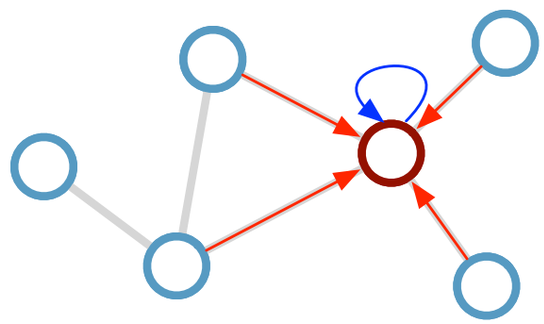


From CNNs to GNNs

Consider this undirected graph:



Calculate update for node in red:



Update rule:

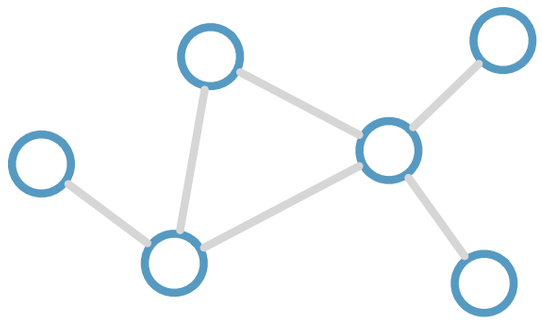
$$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{h}_i^{(l)}\mathbf{W}_0^{(l)} + \sum_{j\in\mathcal{N}_i}\frac{1}{c_{ij}}\mathbf{h}_j^{(l)}\mathbf{W}_1^{(l)}\right)$$

$\mathcal{N}_i$: neighbor indices

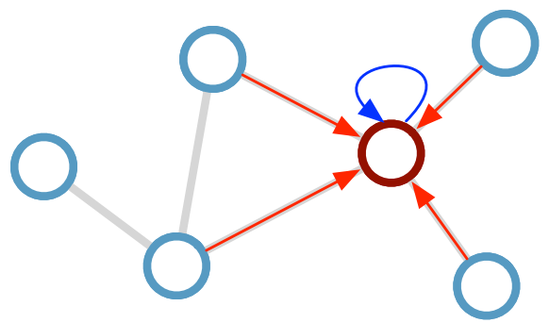
c_{ij} : norm. constant (fixed/trainable)

From CNNs to GNNs

Consider this undirected graph:



Calculate update for node in red:



Update rule:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{h}_i^{(l)}\mathbf{W}_0^{(l)} + \sum_{j\in\mathcal{N}_i}\frac{1}{c_{ij}}\mathbf{h}_j^{(l)}\mathbf{W}_1^{(l)}\right)$$

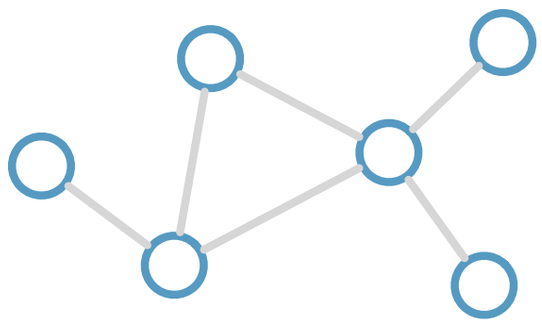
Scalability: subsample messages [Hamilton et al., NIPS 2017]

$\mathcal{N}_i$: neighbor indices

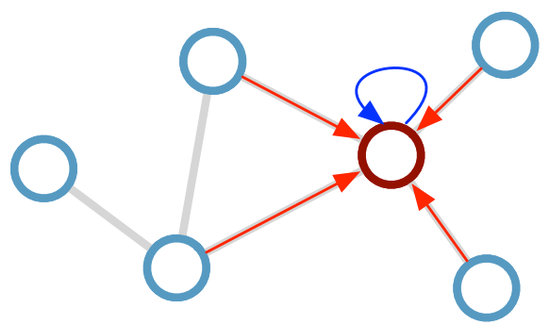
c_{ij} : norm. constant (fixed/trainable)

From CNNs to GNNs

Consider this undirected graph:



Calculate update for node in red:



Desirable properties:

- Weight sharing over all locations
- Invariance to permutations
- Linear complexity O(E)

Update rule:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{h}_i^{(l)}\mathbf{W}_0^{(l)} + \sum_{j\in\mathcal{N}_i}\frac{1}{c_{ij}}\mathbf{h}_j^{(l)}\mathbf{W}_1^{(l)}\right)$$

Scalability: subsample messages [Hamilton et al., NIPS 2017]

$\mathcal{N}_i$: neighbor indices

c_{ij} : norm. constant (fixed/trainable)