Carnegie Mellon
School of Computer Science

Deep Reinforcement Learning and Control

Exploration and Function Approximation

CMU 10-403

Katerina Fragkiadaki

This lecture

Exploration in Large Continuous State Spaces

Exploration: It's all about modelling our uncertainty (again)

- Exploration: trying out new things (new behaviours), with the hope of discovering higher rewards
- Exploitation: doing what **you know** will yield the highest reward
- We explore efficiently once **we know what we do not know,** and target our exploration to the unknown part of the space.
- All non-naive exploration methods consider some form **of uncertainty estimation, regarding policies, Q-functions, or transition dynamics..**

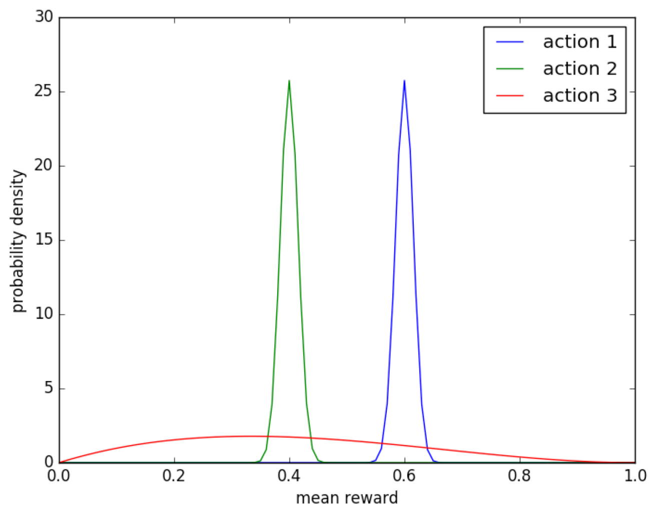
Previous lecture: Exploration in Bandits

- Stateless.
- Q: what does this mean?

Previous lecture: Exploration in Bandits

Thompson Sampling

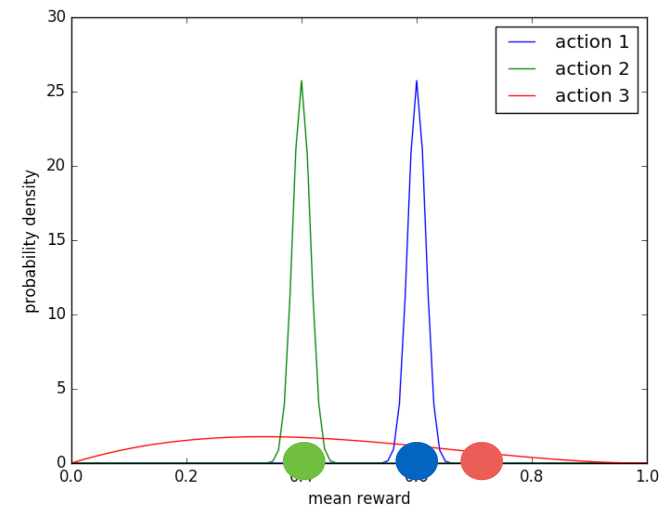
Represent a **posterior distribution** of mean rewards of the arms, as opposed to **point estimates**.



Previous lecture: Exploration in Bandits

Thompson Sampling

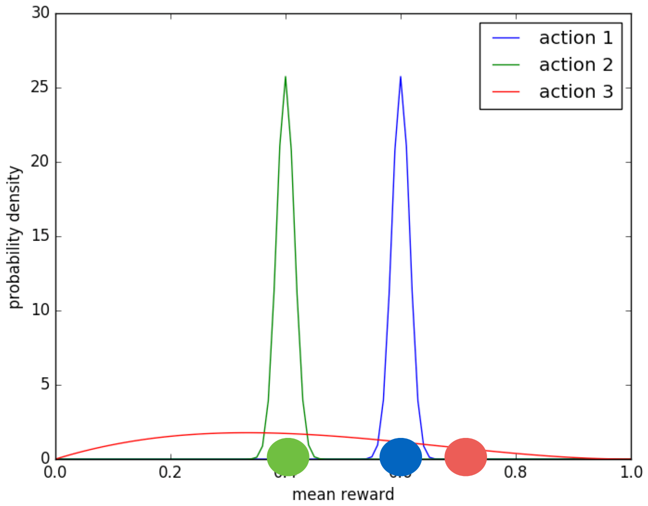
Represent a **posterior distribution** of mean rewards of the arms, as opposed to **point estimates.**



1. Sample from it $\theta_1, \theta_2, \cdots, \theta_k \sim \hat{p}(\theta_1, \theta_2 \cdots \theta_k)$

Previous lecture: Exploration in Bandits

Thompson Sampling



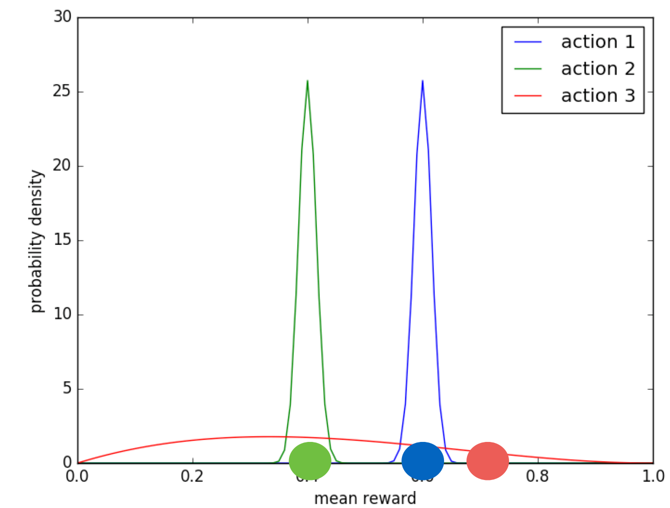
Represent a **posterior distribution** of mean rewards of the arms, as opposed to **point estimates**.

1. Sample from it $\theta_1, \theta_2, \cdots, \theta_k \sim \hat{p}(\theta_1, \theta_2 \cdots \theta_k)$
2. Choose action $a = \arg\max_a \mathbb{E}_\theta[r(a)]$

Previous lecture: Exploration in Bandits

Thompson Sampling

Represent a **posterior distribution** of mean rewards of the arms, as opposed to **point estimates**.



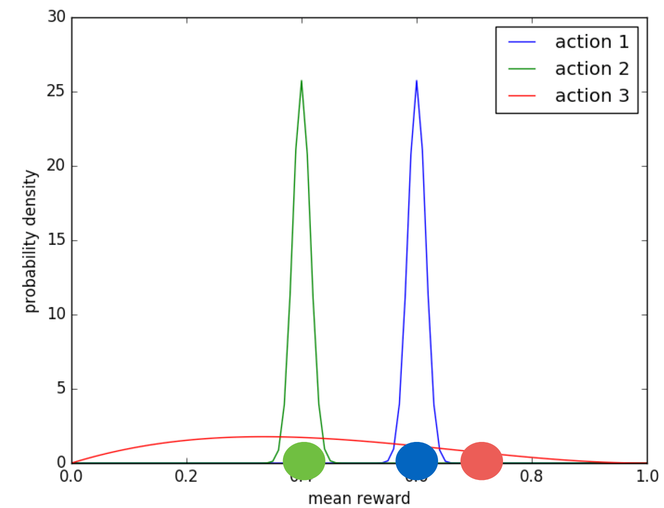
1. Sample from it $\theta_1, \theta_2, \cdots, \theta_k \sim \hat{p}(\theta_1, \theta_2 \cdots \theta_k)$
2. Choose action $a = \arg\max_a \mathbb{E}_\theta[r(a)]$

Play the red arm!

Previous lecture: Exploration in Bandits

Thompson Sampling

Represent a **posterior distribution** of mean rewards of the arms, as opposed to **point estimates**.



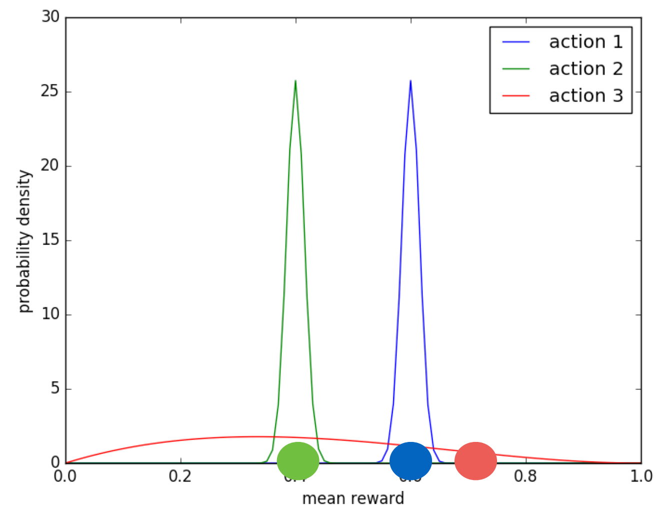
1. Sample from it $\theta_1, \theta_2, \cdots, \theta_k \sim \hat{p}(\theta_1, \theta_2 \cdots \theta_k)$
2. Choose action $a = \arg\max_a \mathbb{E}_\theta[r(a)]$
3. Play action, observe reward

0.8!

Previous lecture: Exploration in Bandits

Thompson Sampling

Represent a **posterior distribution** of mean rewards of the arms, as opposed to **point estimates**.



1. Sample from it $\theta_1, \theta_2, \cdots, \theta_k \sim \hat{p}(\theta_1, \theta_2 \cdots \theta_k)$
2. Choose action $a = \arg\max_a \mathbb{E}_\theta[r(a)]$
3. Play action, observe reward
4. Update the mean reward distribution

- Can I do something like that for general MDPs?
- What is the equivalent of mean rewards for geenral MDP?

Exploration via Posterior Sampling of Q functions

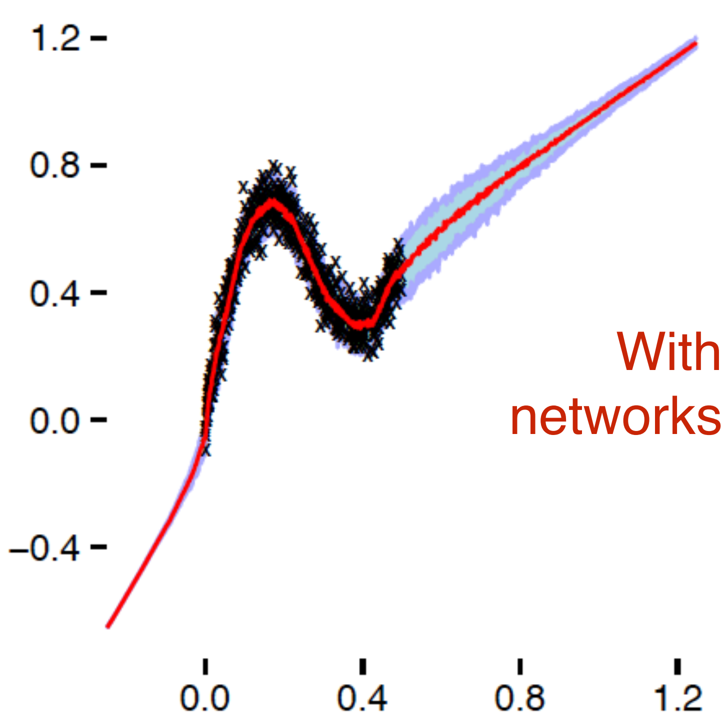
Represent a posterior distribution of Q functions, instead of a point estimate.

1. Sample from P(Q) $Q \sim P(Q)$
2. Choose actions according to this Q for one episode $a = \arg\max_a Q(a, s)$
3. Update the Q distribution using the collected experience tuples

Then we do not need \epsilon-greedy for exploration! Better exploration by representing uncertainty over Q.

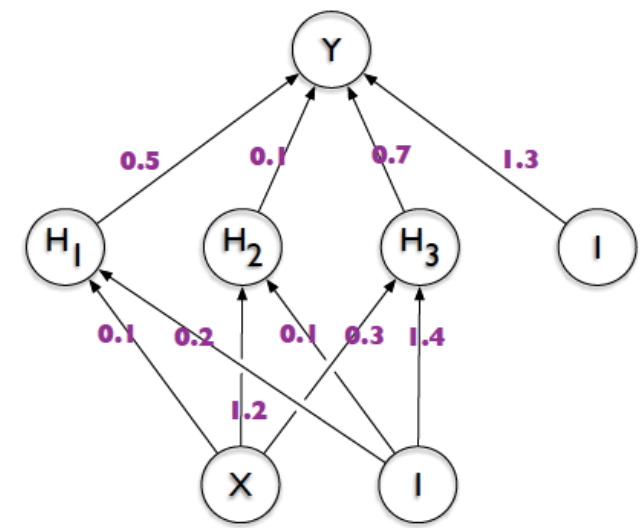
But how can we learn a distribution of Q functions P(Q) if Q function is a deep neural network?

Representing Uncertainty in Deep Learning

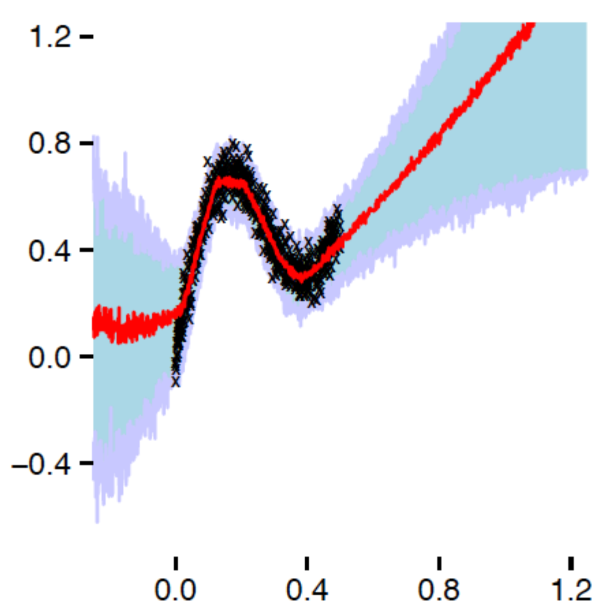


With standard regression networks we cannot represent our uncertainty

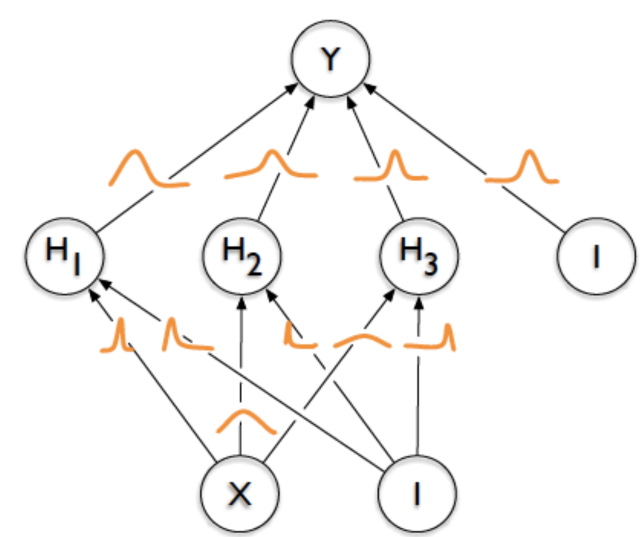
A regression network trained on **X**



$$\mathbf{w}^{\mathrm{MAP}} = \arg\max_{\mathbf{w}} \log P(\mathbf{w}|\mathcal{D})$$
$$= \arg\max_{\mathbf{w}} \log P(\mathcal{D}|\mathbf{w}) + \log P(\mathbf{w}).$$



A bayesian regression network trained on **X**



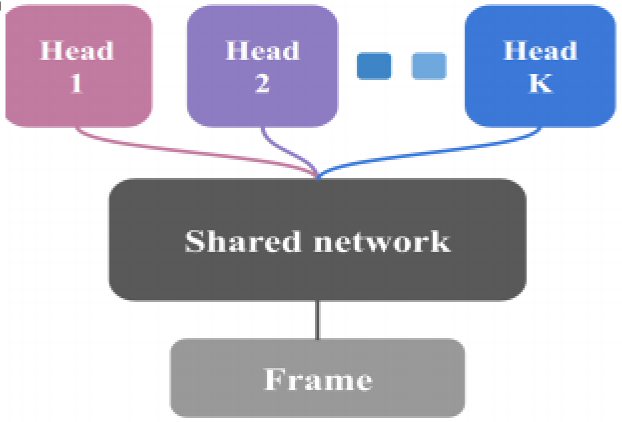
$P(\mathbf{w}\,|\,\mathscr{D})$

Exploration via Posterior Sampling of Q-functions

1. **Bayesian neural networks**. Estimate posteriors for the neural weights, as opposed to point estimates. We just saw that..

2. **Neural network ensembles.** Train multiple Q-function approximations each on using different subset of the data. A reasonable approximation to 1.

3. **Neural network ensembles with shared backbone**. Only the heads are trained with different subset of the data. A reasonable approximation to 2 with less computation.



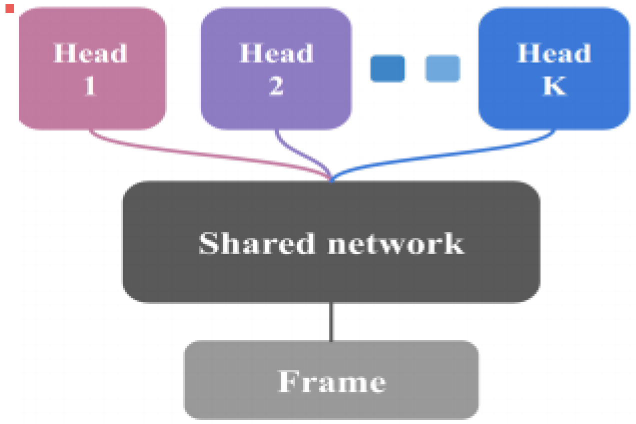
4. **Ensembling by dropout.** Randomly mask-out (zero out)neural network weights, to create different neural nets, both at train and test time. reasonable approximation to 2.

Exploration via Posterior Sampling of Q-functions

1. **Bayesian neural networks**. Estimate posteriors for the neural weights, as opposed to point estimates. We just saw that..

2. **Neural network ensembles.** Train multiple Q-function approximations each on using different subset of the data. A reasonable approximation to 1.

3. **Neural network ensembles with shared backbone**. Only the heads are trained with different subset of the data. A reasonable approximation to 2 with less computation.

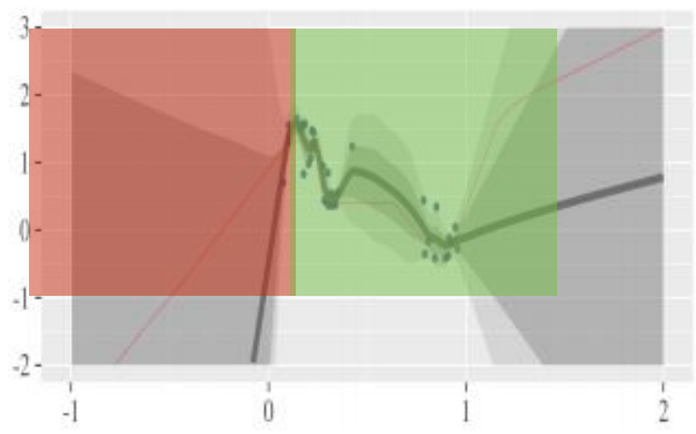


4. **Ensembling by dropout.** Randomly mask-out (zero out)neural network weights, to create different neural nets, both at train and test time. reasonable approximation to 2. (but authors showed 3. worked better than 4.)

Deep exploration with bootstrapped DQN, Osband et al.

Exploration via Posterior Sampling of Q-functions

1. Sample from P(Q) $Q \sim P(Q)$
2. Choose actions according to this Q for one episode $a = \arg\max_a Q(a, s)$
3. Update the Q distribution using the collected experience tuples

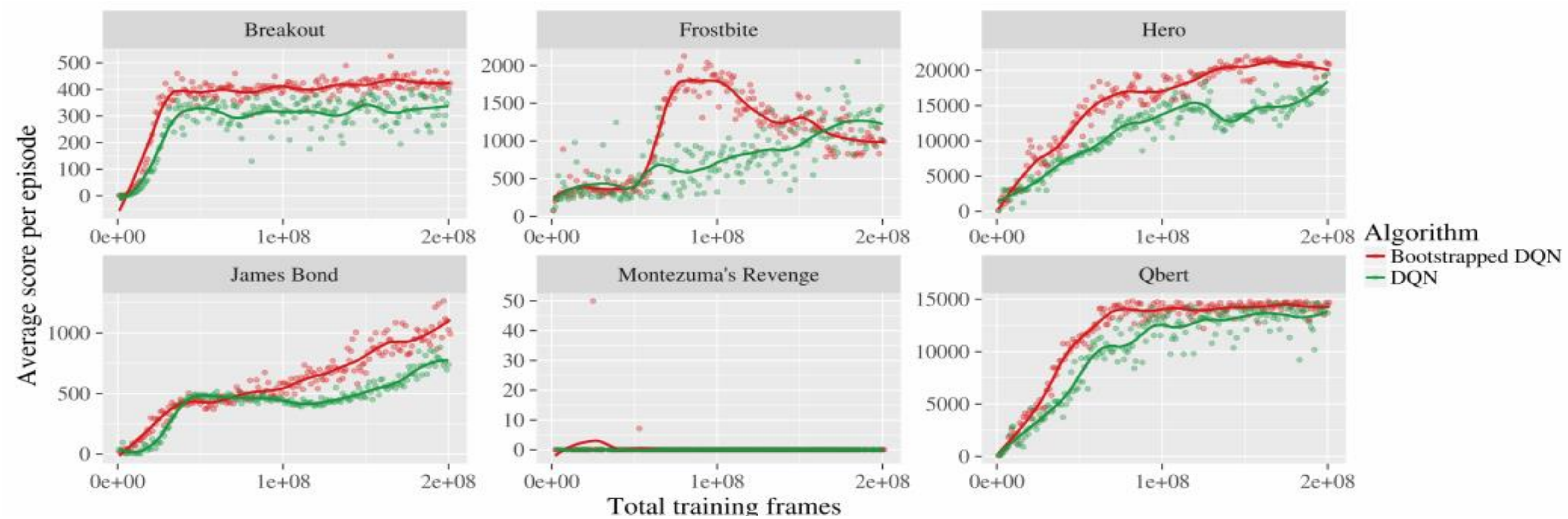


With ensembles we achieve similar things as with Bayesian nets:

- The entropy of predictions of the network (obtained by sampling different heads) is high in the no data regime. Thus, Q function values will have high entropy there and encourage exploration.
- When Q values have low entropy, i exploit, i do not explore.

Deep exploration with bootstrapped DQN, Osband et al.

Exploration via Posterior Sampling of Q-functions



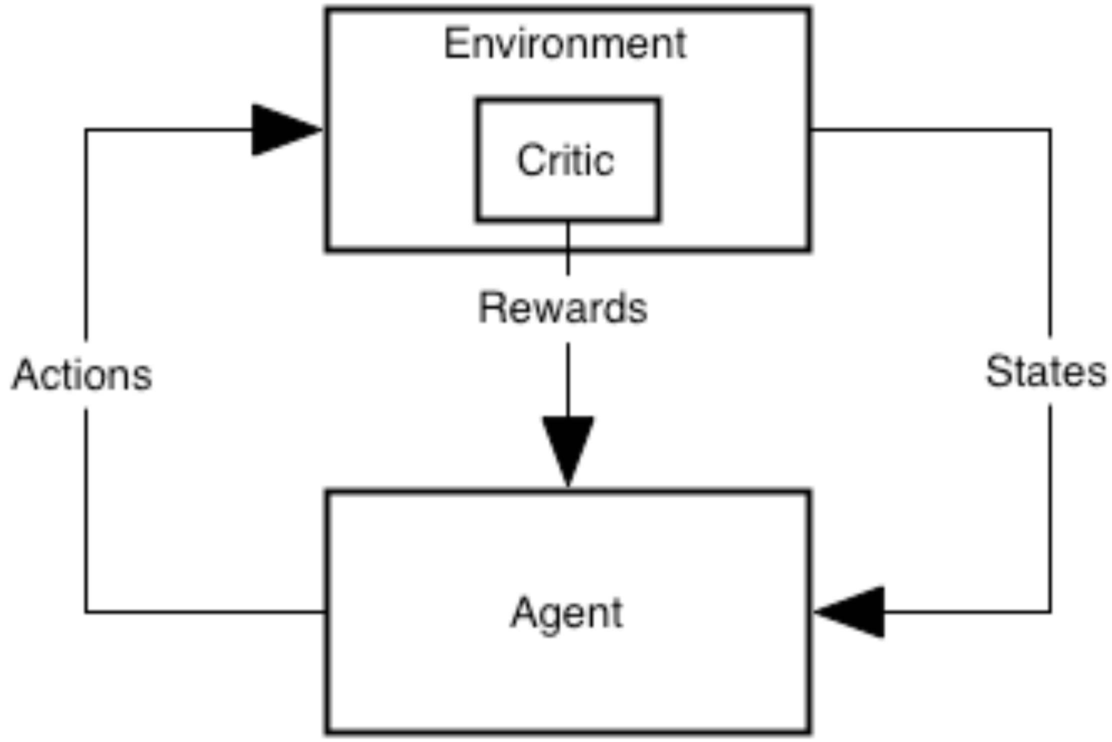
Deep exploration with bootstrapped DQN, Osband et al.

Motivation

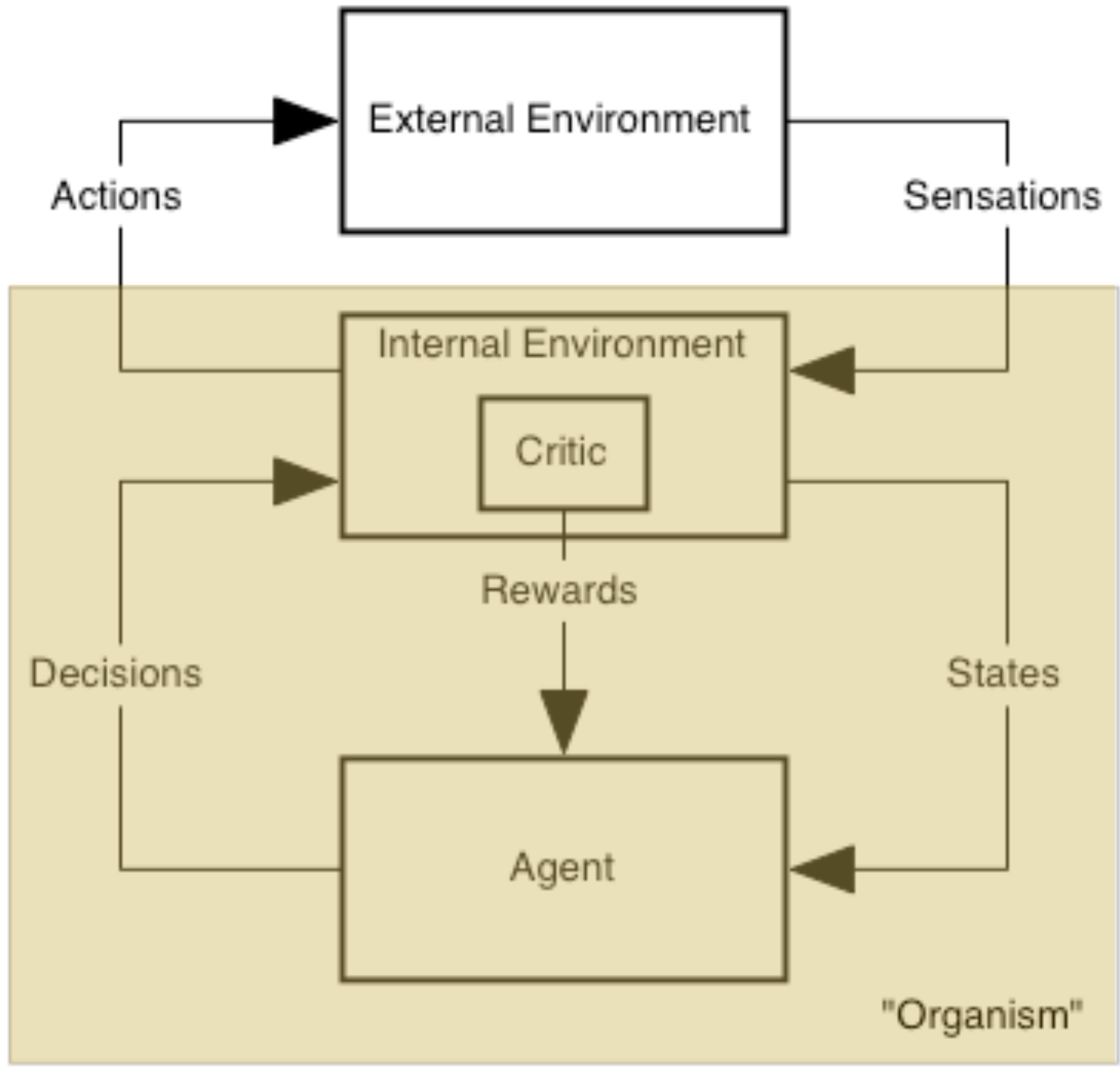
Motivation: “Forces” that energize an organism to act and that direct its activity

- Extrinsic Motivation: being moved to do something because of some external reward ($$, a prize, etc.).
    - Problem: such rewards are sparse..
- Intrinsic Motivation: being moved to do something because it is inherently enjoyable (curiosity, exploration, novelty, surprise, incongruity, complexity…)
    - Gain: Task independent! Free of human supervision, no need to code up reward functions to incentivize the agent. A general loss functions that drives learning.

Extrinsic Rewards



Intrinsic Rewards



All rewards are intrinsic

Curiosity VS Survival

"As knowledge accumulated about the conditions that govern exploratory behavior and about how quickly it appears after birth, it seemed less and *less likely that this behavior could be a derivative of hunger, thirst, sexual appetite, pain, fear of pain, and the like*, or that stimuli sought through exploration are welcomed because they have previously accompanied satisfaction of these drives."

D. E. Berlyne, *Curiosity and Exploration*, Science, 1966

Intrinsic Motivation different than Intrinsic Necessity: being moved to do something because it is necessary (eat, drink, find shelter from rain…)

Curiosity and Never-ending Learning

Why should we care?

- Because curiosity is a general, task independent cost function, that if we successfully incorporate to our learning machines, it may result in agents that (want to) improve with experience, like people do.
- Those intelligent agents would not require supervision by coding up reward functions for every little task, they would learn (almost) autonomously
- Curiosity-driven motivation is beyond satisfaction of hunger, thirst, and other biological activities (which arguably would be harder to code up in artificial agents..)

Curiosity-driven exploration

Seek **novelty/surprise**:

- Visit *novel states* s
- Observe *novel state transitions* (s,a)->s'

Q: How can we computationally formalize that?

Curiosity-driven exploration-one way to do it

We will add exploration reward bonuses to the extrinsics (task-related) rewards:

Independent of the task in hand!

$$R^t(s,a,s') = \underbrace{r(s,a,s')}_{extrinsic} + \underbrace{\mathscr{B}^t(s,a,s')}_{intrinsic}$$

We would then be using rewards $R^t(s,a,s')$ in our favorite model free RL method.

Curiosity-driven exploration-one way to do it

We will add exploration reward bonuses to the extrinsics (task-related) rewards:

Independent of the task in hand!

$$R^t(s, a, s') = \underbrace{r(s, a, s')}_{extrinsic} + \underbrace{\mathscr{B}^t(s, a, s')}_{intrinsic}$$

We would then be using rewards $R^t(s, a, s')$ in our favorite model free RL method.

Exploration reward bonuses are non stationary: as the agent interacts with the environment, what is now new and novel, becomes old and known.

Curiosity-driven exploration

Seek **novelty/surprise**:

- Visit *novel states* s
- Observe *novel state transitions* (s,a)->s’

State Visitation counts in Small MDPs

Book-keep state visitation counts $N(s)$

Add exploration reward bonuses that encourage policies that visit states with fewer counts.

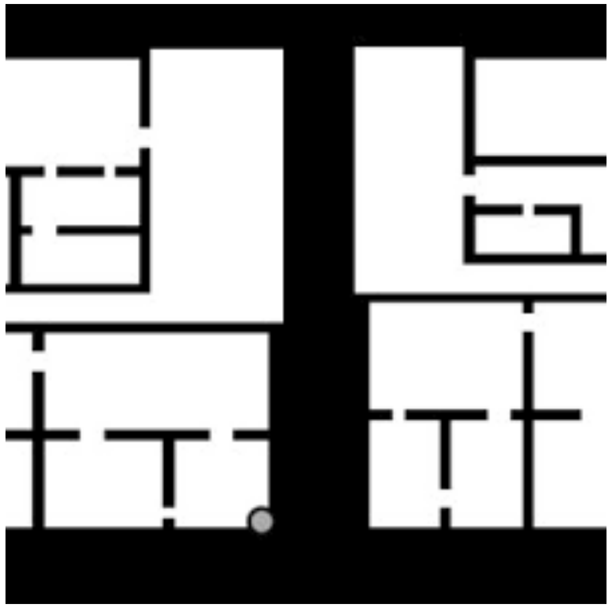
$$R^t(s, a, s') = \underbrace{r(s, a, s')}_{extrinsic} + \underbrace{\mathcal{B}(N(s))}_{intrinsic}$$

$N(s)$: number of times i visited state s

UCB: $\mathcal{B}(N(\mathbf{s})) = \sqrt{\frac{2 \ln n}{N(\mathbf{s})}}$

MBIE-EB (Strehl & Littman, 2008): $\mathcal{B}(N(\mathbf{s})) = \sqrt{\frac{1}{N(\mathbf{s})}}$

BEB (Kolter & Ng, 2009): $\mathcal{B}(N(\mathbf{s})) = \frac{1}{N(\mathbf{s})}$



State Visitation Counts in High Dimensions

- We want to come up with something that rewards states that we have not visited often.
- But in high dimensions, we rarely visit a state twice!
- We need to capture a notion of state similarity, and reward states that are most *dissimilar* that what we have seen so far, as opposed to *different* (as they will always be different)

$$R^t(s,a,s') = \underbrace{r(s,a,s')}_{extrinsic} + \underbrace{\mathscr{B}(N(s))}_{intrinsic}$$

the rich natural world

Unifying Count-Based Exploration and Intrinsic Motivation

Marc G. Bellemare
bellemare@google.com

Sriram Srinivasan
srsrinivasan@google.com

Georg Ostrovski
ostrovski@google.com

Tom Schaul
schaul@google.com

David Saxton
saxton@google.com

Rémi Munos
munos@google.com

- We use parametrized density estimates instead of discrete counts.
- $p_\theta(s)$:parametrized visitation density: how much we have visited state s.
- Even if we have not seen exactly the same state s, the probability can be high if we visited similar states.

Exploring with Pseudcounts

State s

fit model $p_\theta(\mathbf{s})$ to all states $\mathcal{D}$ seen so far

take a step i and observe $\mathbf{s}_i$

fit new model $p_{\theta'}(\mathbf{s})$ to $\mathcal{D} \cup \mathbf{s}_i$

use $p_\theta(\mathbf{s}_i)$ and $p_{\theta'}(\mathbf{s}_i)$ to estimate $\hat{N}(\mathbf{s})$

set $r_i^+ = r_i + \mathcal{B}(\hat{N}(\mathbf{s}))$ ← "pseudo-count"

$$\mathcal{B}(N(\mathbf{s})) = \sqrt{\frac{1}{N(\mathbf{s})}}$$

how to get $\hat{N}(\mathbf{s})$? use the equations

$$p_\theta(\mathbf{s}_i) = \frac{\hat{N}(\mathbf{s}_i)}{\hat{n}} \qquad p_{\theta'}(\mathbf{s}_i) = \frac{\hat{N}(\mathbf{s}_i) + 1}{\hat{n} + 1}$$

two equations and two unknowns!

$$\hat{N}(\mathbf{s}_i) = \hat{n} p_\theta(\mathbf{s}_i) \qquad \hat{n} = \frac{1 - p_{\theta'}(\mathbf{s}_i)}{p_{\theta'}(\mathbf{s}_i) - p_\theta(\mathbf{s}_i)} p_\theta(\mathbf{s}_i)$$

Unifying Count-Based Exploration and Intrinsic Motivation, Bellemare et al.

Exploring with Pseudcounts

State s

fit model $p_\theta(\mathbf{s})$ to all states $\mathcal{D}$ seen so far

take a step i and observe $\mathbf{s}_i$

fit new model $p_{\theta'}(\mathbf{s})$ to $\mathcal{D} \cup \mathbf{s}_i$

use $p_\theta(\mathbf{s}_i)$ and $p_{\theta'}(\mathbf{s}_i)$ to estimate $\hat{N}(\mathbf{s})$

set $r_i^+ = r_i + \mathcal{B}(\hat{N}(\mathbf{s}))$ ← "pseudo-count"

how to get $\hat{N}(\mathbf{s})$? use the equations

$$p_\theta(\mathbf{s}_i) = \frac{\hat{N}(\mathbf{s}_i)}{\hat{n}} \qquad p_{\theta'}(\mathbf{s}_i) = \frac{\hat{N}(\mathbf{s}_i) + 1}{\hat{n} + 1}$$

two equations and two unknowns!

$$\mathcal{B}(N(\mathbf{s})) = \sqrt{\frac{1}{N(\mathbf{s})}}$$

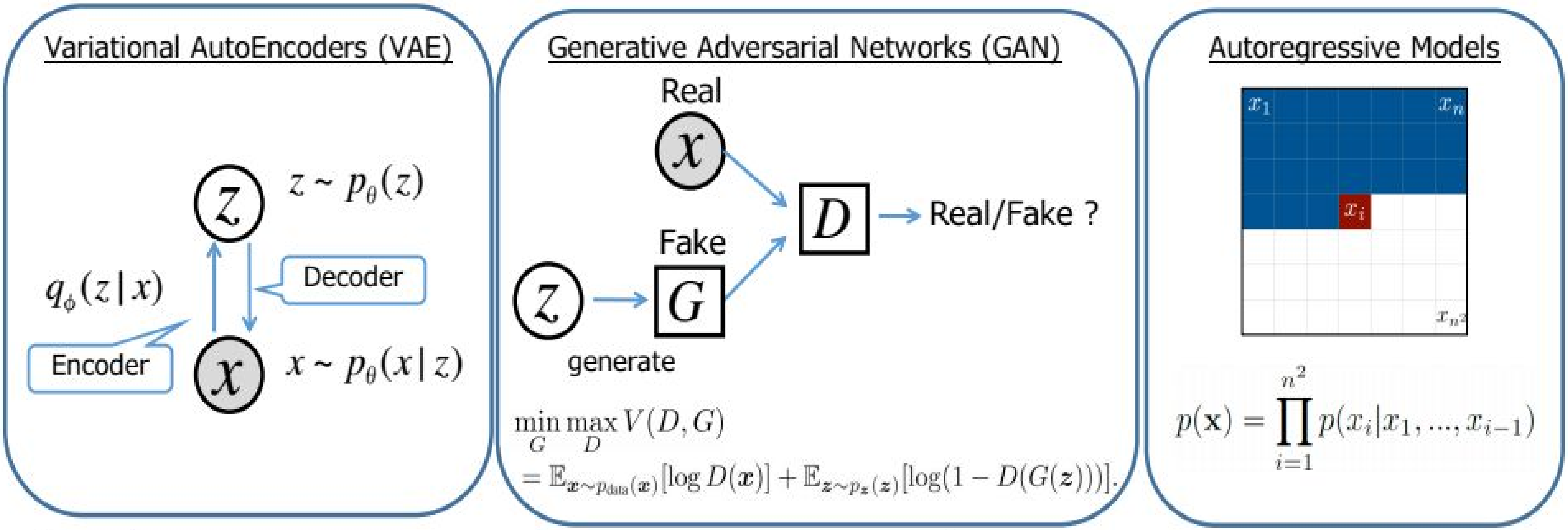
$$\hat{N}(\mathbf{s}_i) = \hat{n} p_\theta(\mathbf{s}_i) \qquad \hat{n} = \frac{1 - p_{\theta'}(\mathbf{s}_i)}{p_{\theta'}(\mathbf{s}_i) - p_\theta(\mathbf{s}_i)} p_\theta(\mathbf{s}_i)$$

How are we going to estimate $p_\theta(s)$?

A model that given an image predicts a probability: how much I have seen this image in the past.

Unifying Count-Based Exploration and Intrinsic Motivation, Bellemare et al.

Generative models of Images



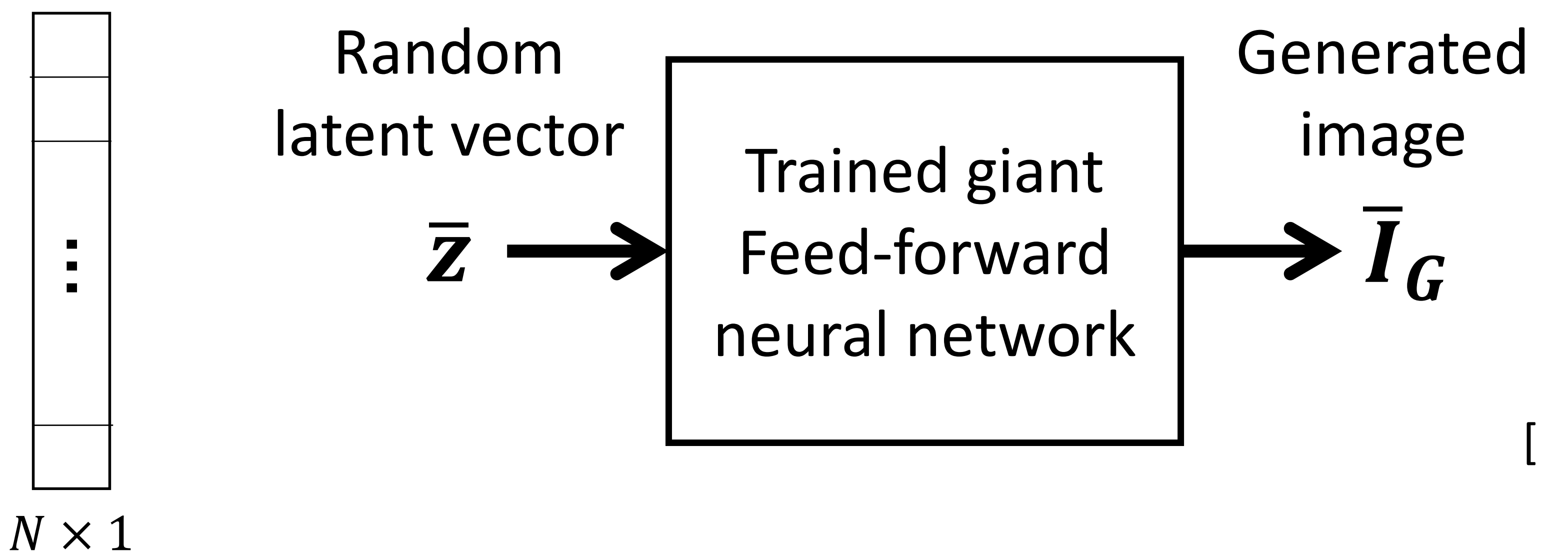
	VAE	GAN	Autoregressive Models
Pros.	- Efficient inference with approximate latent variables.	- generate sharp image. - no need for any Markov chain or approx networks during sampling.	- very simple and stable training process - currently gives the best log likelihood. - tractable likelihood
Cons.	- generated samples tend to be blurry.	- difficult to optimize due to unstable training dynamics.	- relatively inefficient during sampling

(cf. https://openai.com/blog/generative-models/)

We like that! We want it to compute probabilities, not to draw beautiful samples!

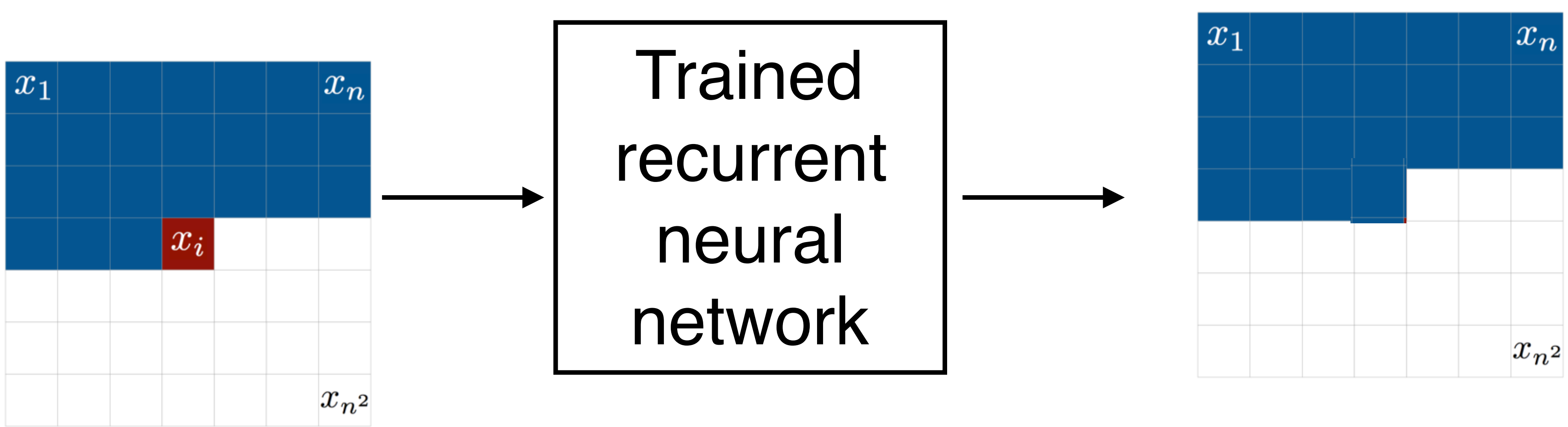
Generative models of Images

- One shot image generation (usually used in VAEs and GANs):



[I. Goodfellow, 2016]

- Autoregressive image generation: Generate the image one pixel at a time



Autoregressive Image generation

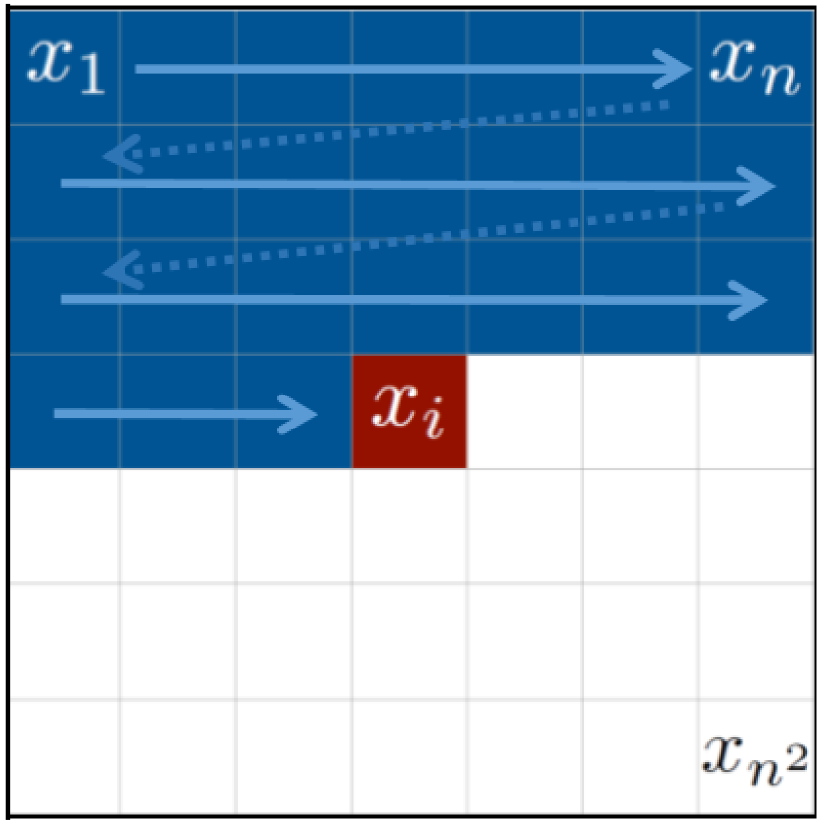
Intuition

$$p(\mathbf{x}) = p(x_1, x_2, ..., x_{n^2})$$

Bayes Theorem:

$$p(\mathbf{x}) = \prod_{i=1}^{n^2} p(x_i | x_1, ..., x_{i-1})$$

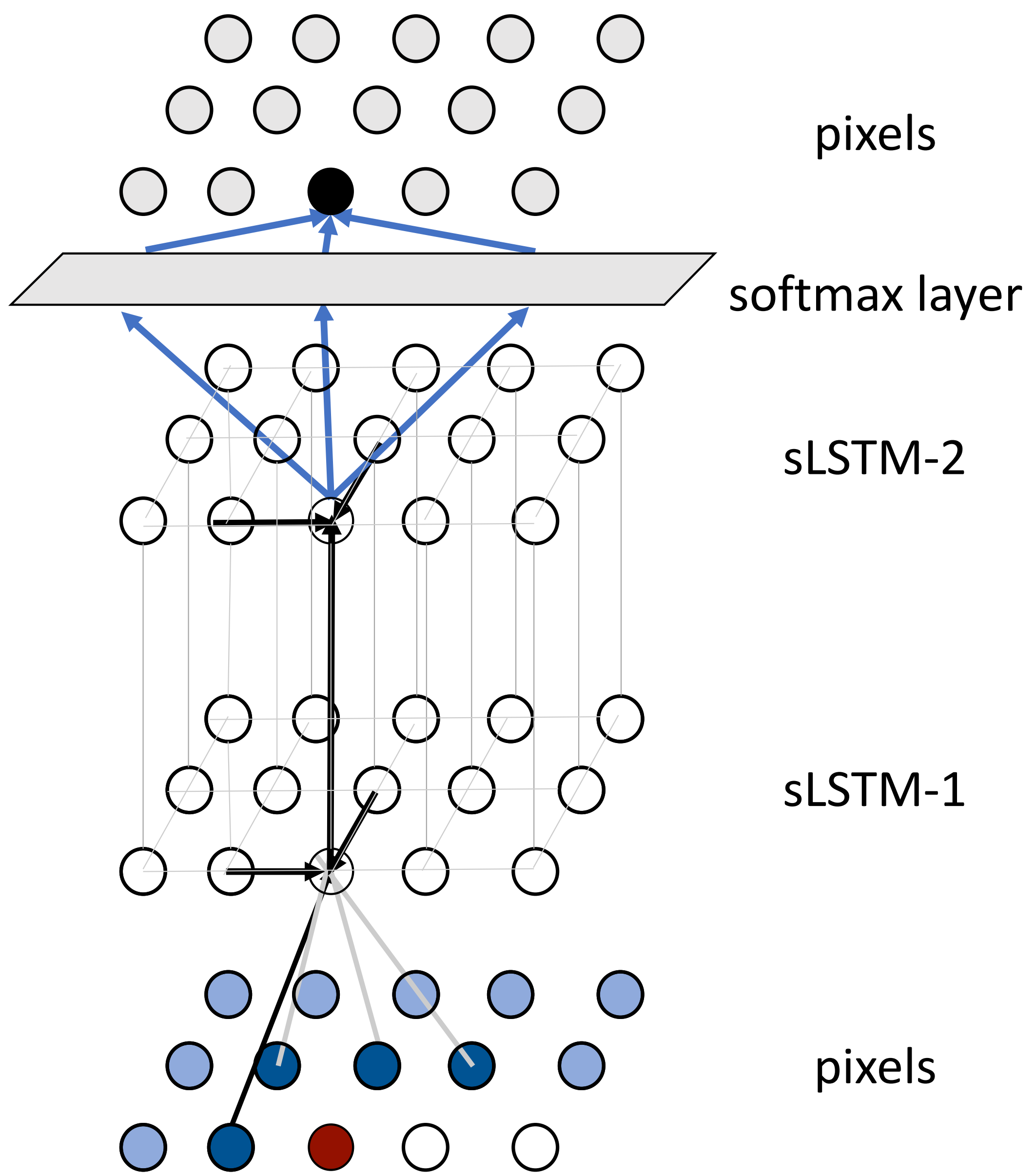
A sequential model!



Pixel recurrent neural networks, ICML 2016

Pixel RNN: a neural networks that sequentially predicts the pixels in the image

Spatial LSTM



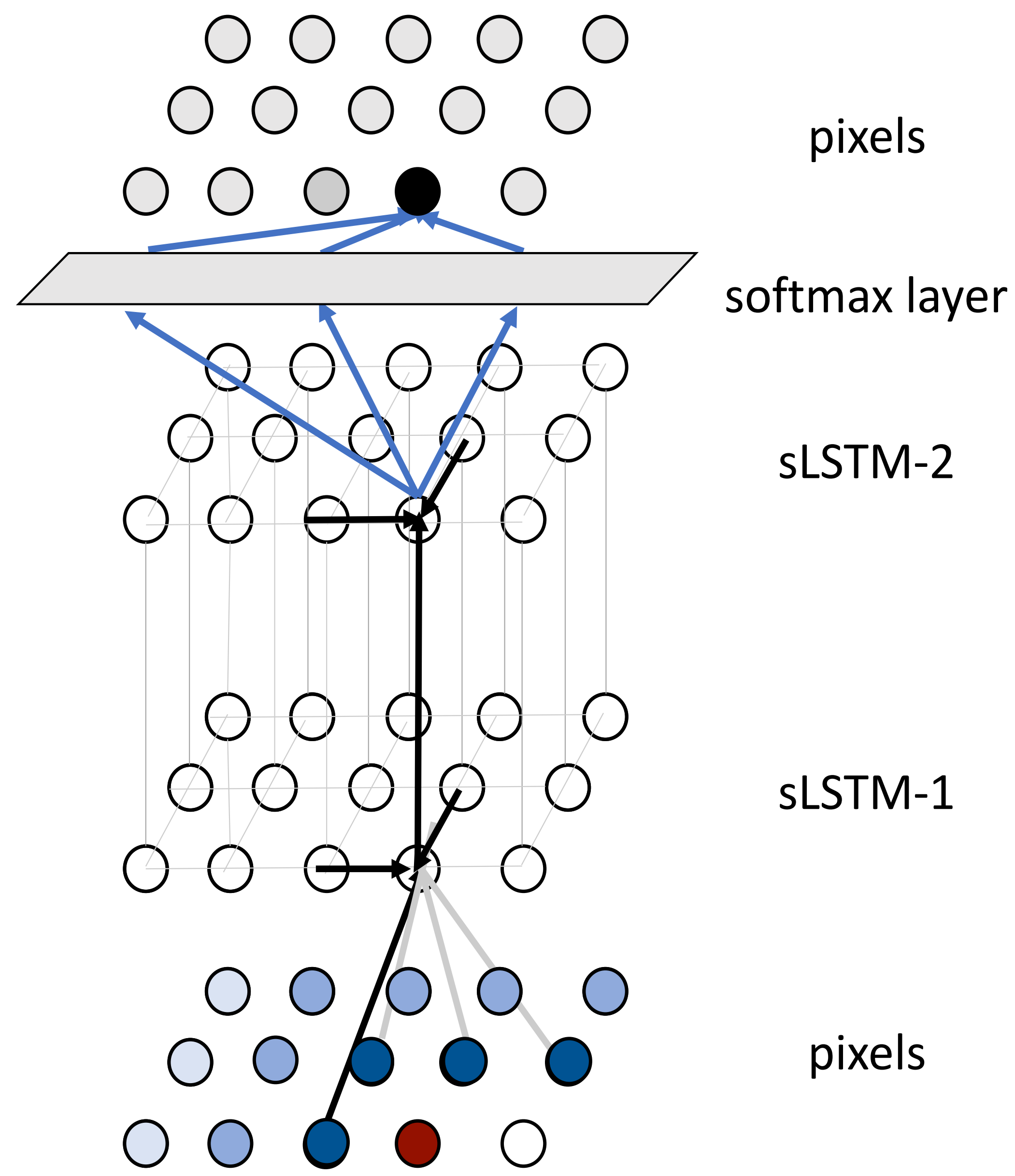
Adapted from: Generative image modeling using spatial LSTM. Theis & Bethge, 2015

x_{ij} the pixel i am estimating the value for

$\mathbf{x}_{<ij}$ the pixel that have already been predicted, and on which our LSTM is conditioning

Spatial LSTM

Too slow, no parallelization: I update the pixels one by one.



Adapted from: Generative image modeling using spatial LSTM. Theis & Bethge, 2015

x_{ij} the pixel i am estimating the value for

$\mathbf{x}_{<ij}$ the pixel that have already been predicted, and on which our LSTM is conditioning

Multinomial Distribution for Pixel Value

- Treat pixels as discrete variables:
  - To estimate a pixel value, do classification in every channel (256 classes indicating pixel values 0-255)
  - Implemented with a final softmax layer

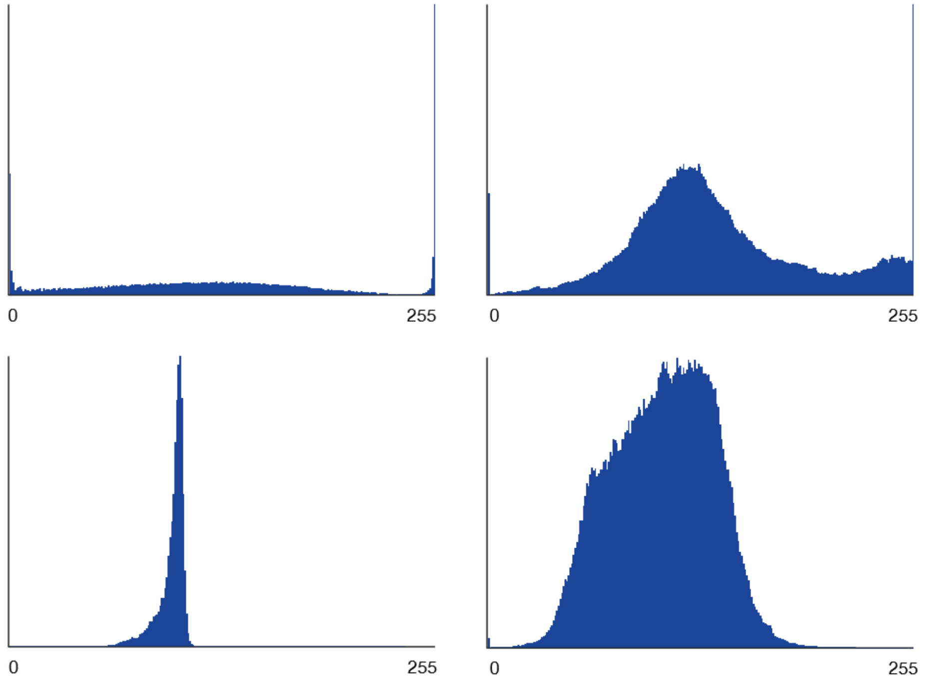
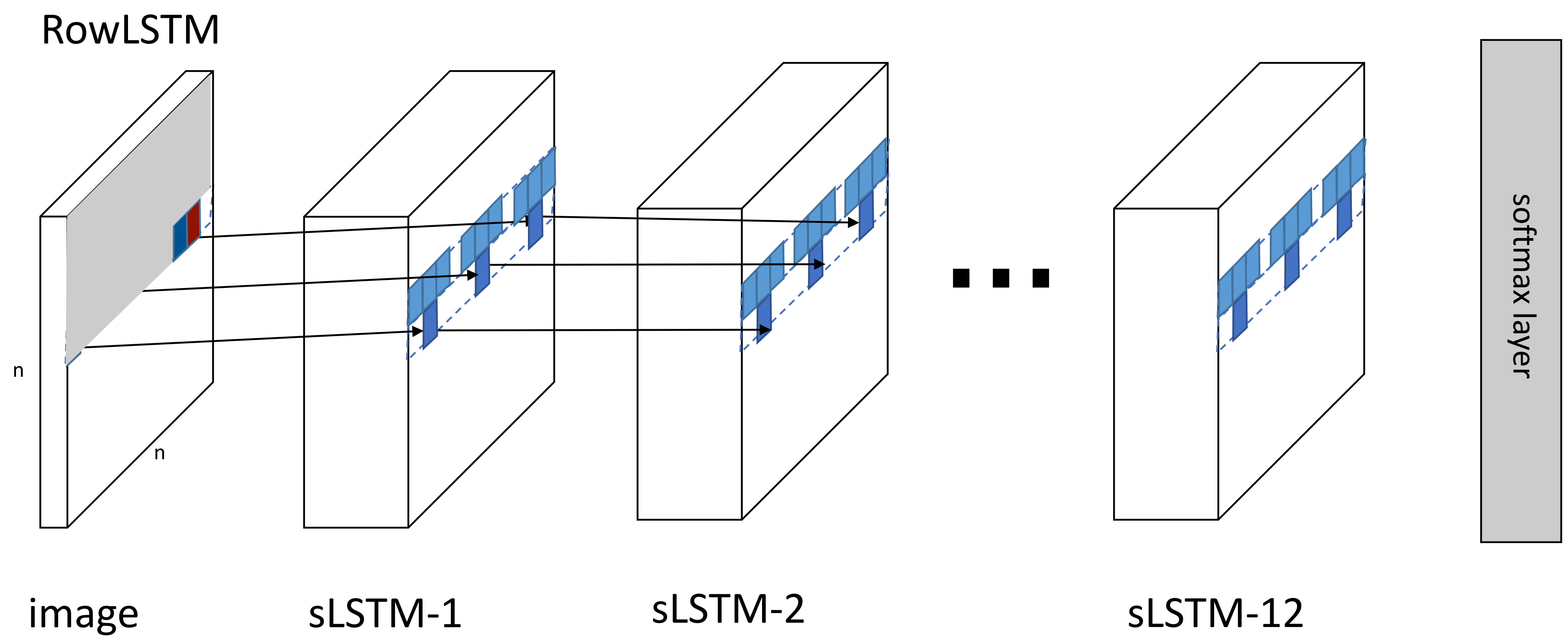


Figure: Example softmax outputs in the final layer, representing probability distribution over 256 classes.

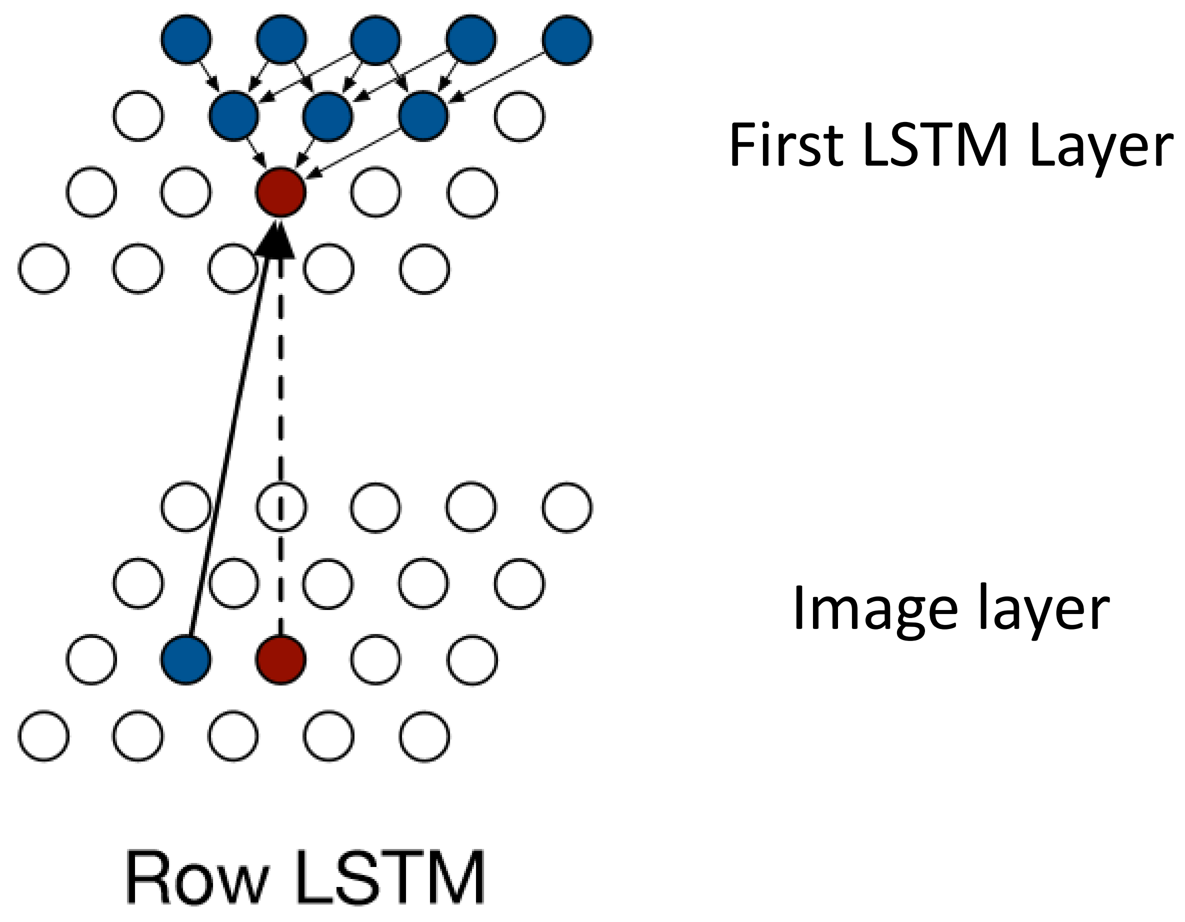
Figure from: [Oord et al.](#)

Pixel RNN



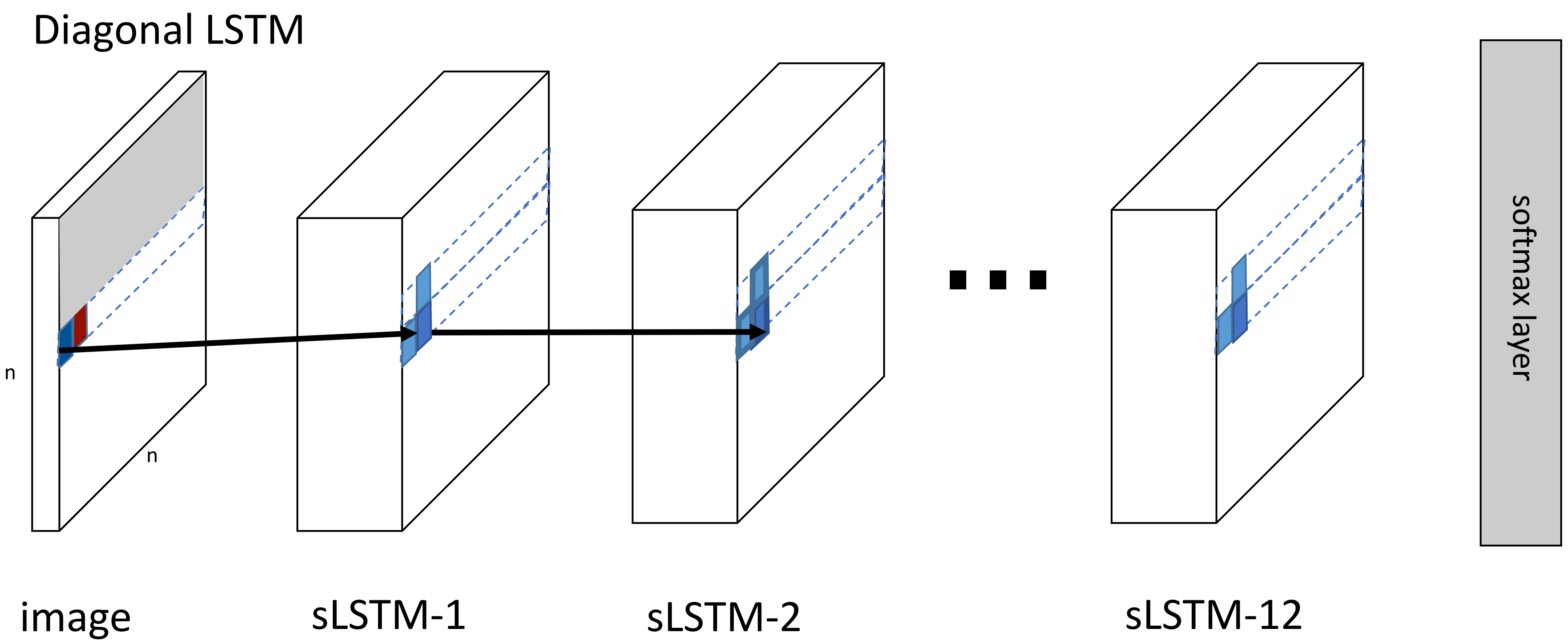
Pixel recurrent neural networks, ICML 2016

Row LSTM



Pixel recurrent neural networks, ICML 2016

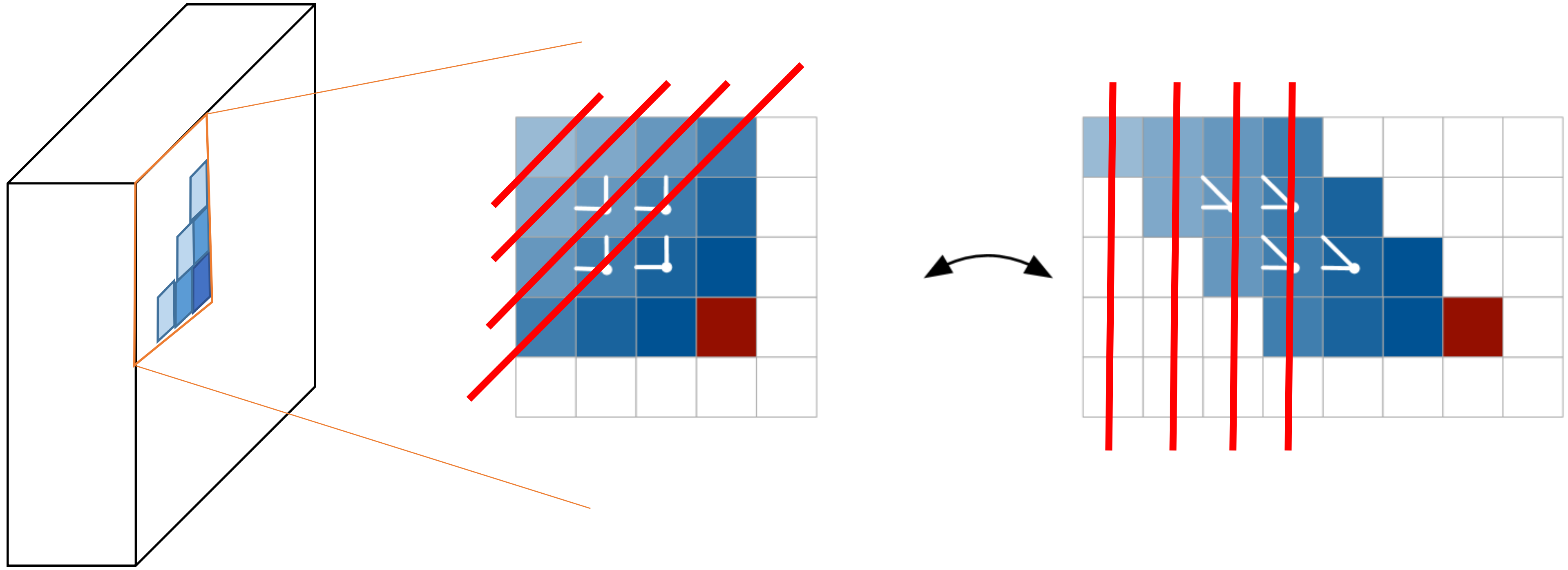
Pixel RNN



Pixel recurrent neural networks, ICML 2016

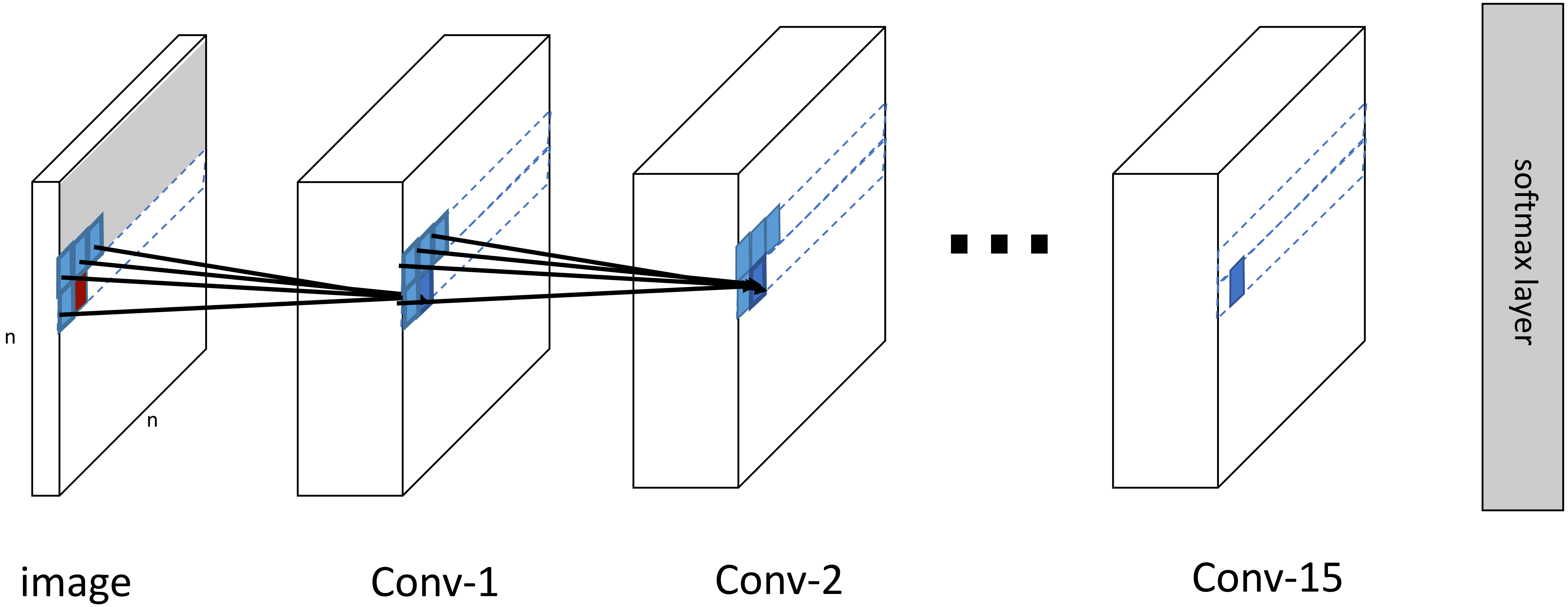
Diagonal LSTM

- To optimize, we skew the feature maps so it can be parallelized



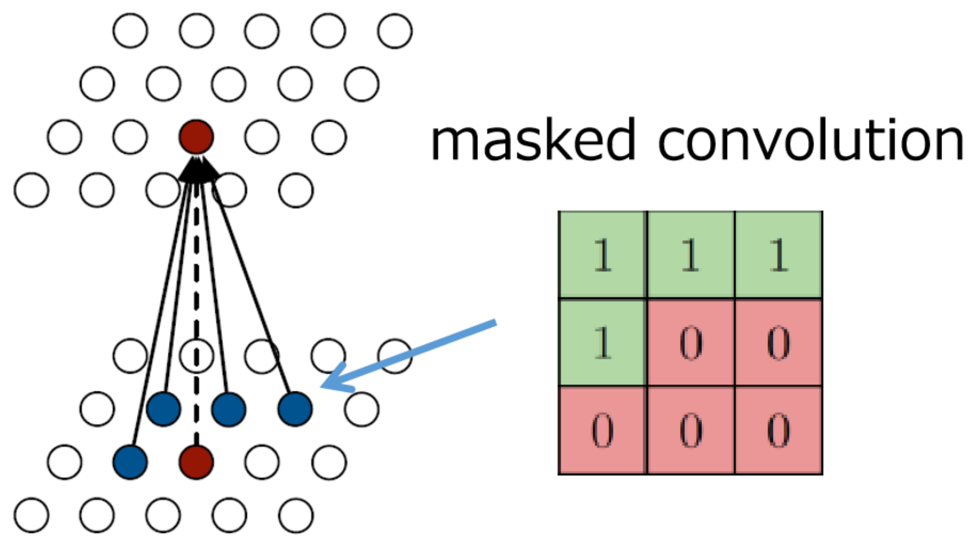
Pixel recurrent neural networks, ICML 2016

Pixel CNN



Pixel CNN

- 2D convolution on previous layer
- Apply masks so a pixel does not see future pixels (in sequential order)



Pixel recurrent neural networks, ICML 2016

Comparison

PixelCNN	PixelRNN – Row LSTM	PixelRNN – Diagonal BiLSTM
Full dependency field	Triangular receptive field	Full dependency field
Fastest	Slow	Slowest
Worst log-likelihood	-	Best log-likelihood

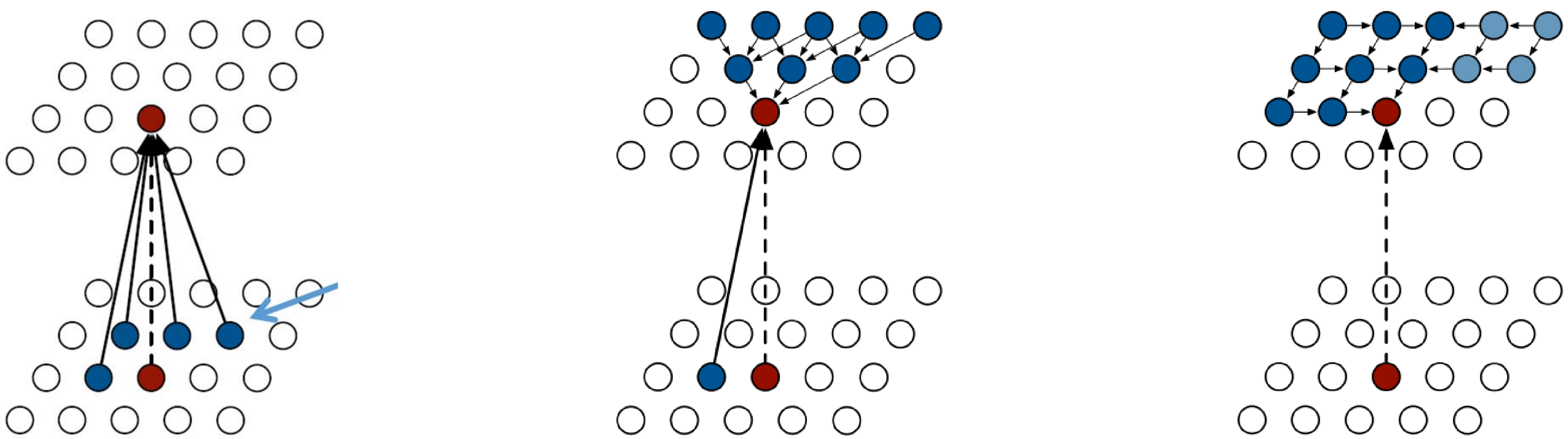
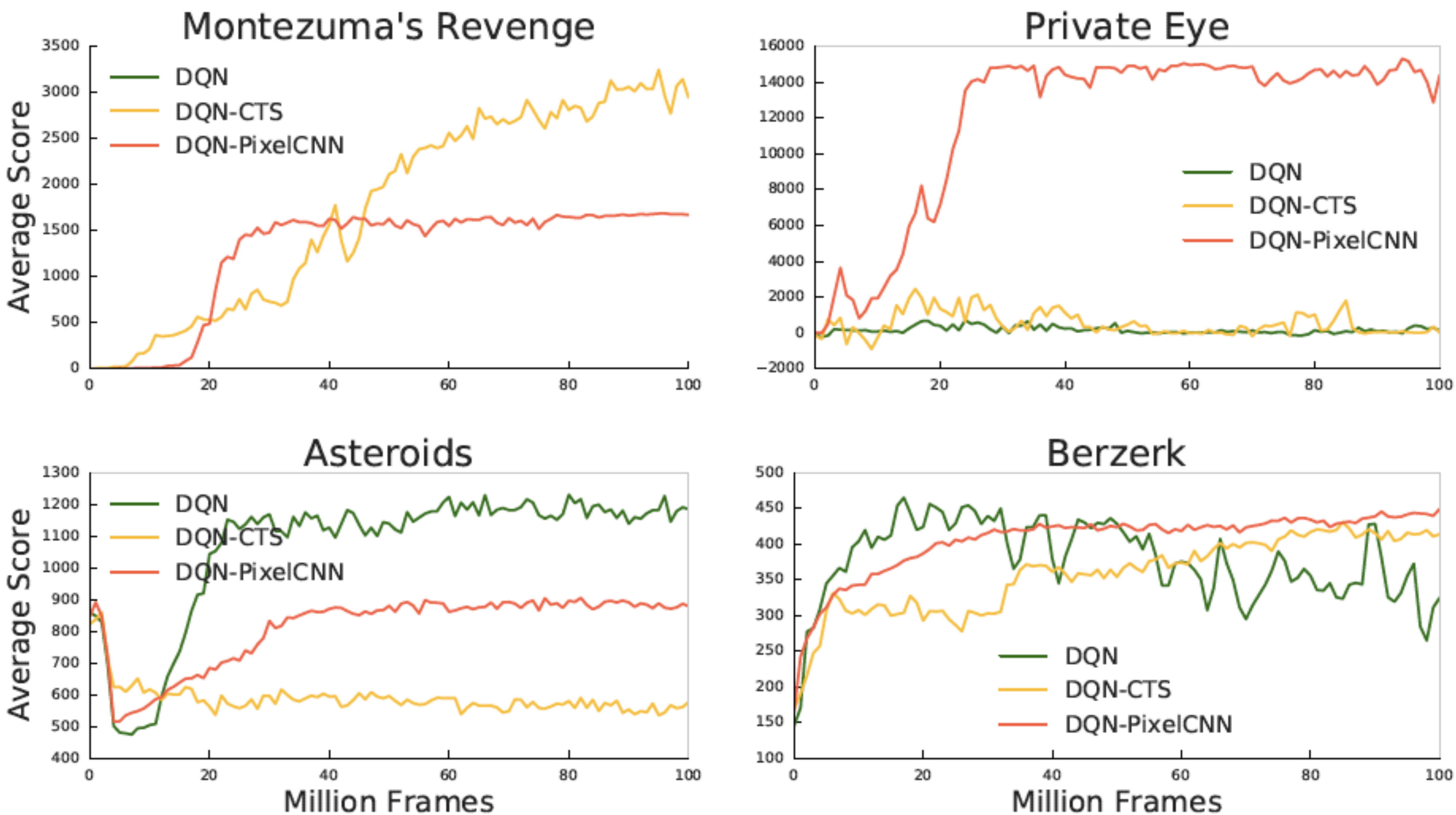


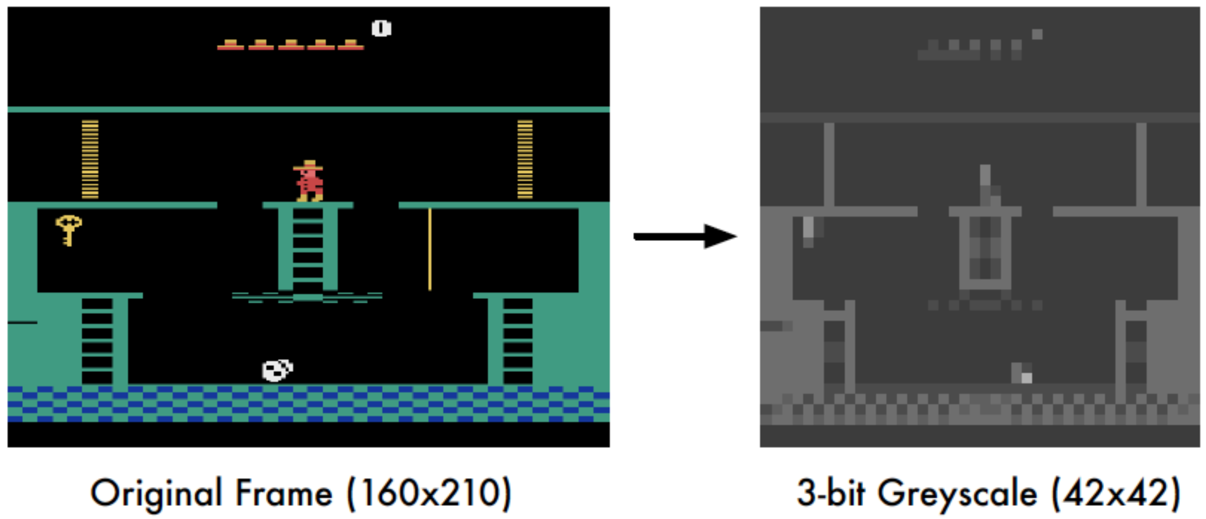
Figure from: [Oord et al.]

Better density estimation usually helps



Better density estimation helps

Frame preprocessing: shrink and convert to grayscale



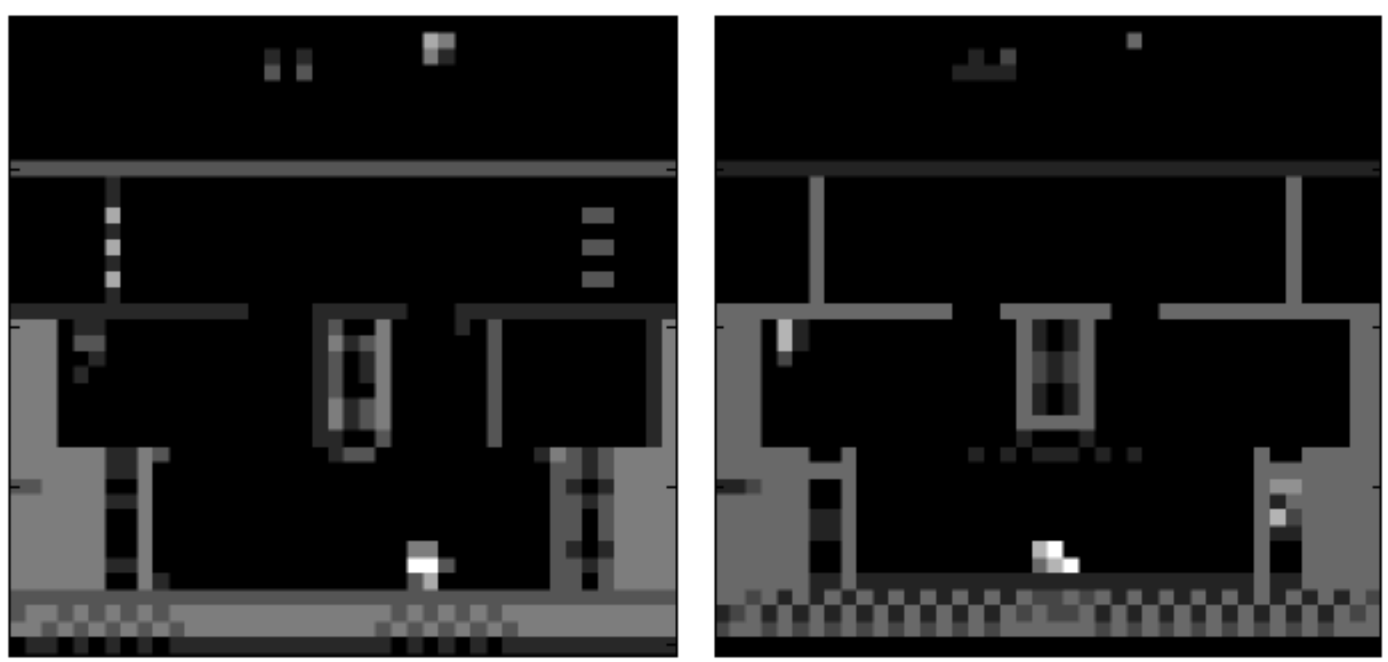


Figure 4. Samples after 25K steps. **Left**: CTS, **right**: PixelCNN.

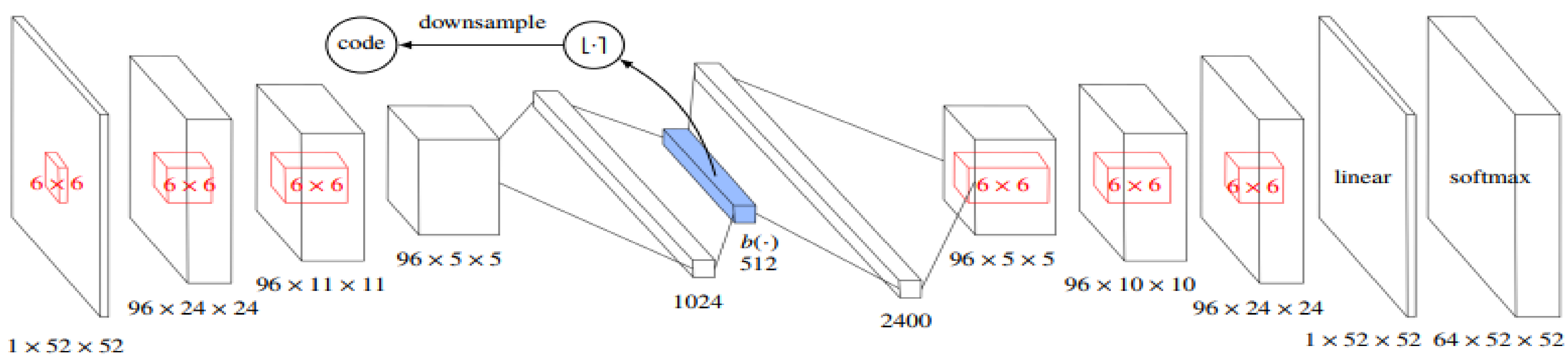
Curiosity-driven exploration

Seek **novelty/surprise**:

- Visit *novel states* s
- Observe *novel state transitions* (s,a)->s’

State Counting with DeepHashing

- We still count states (images) but not in pixel space, but in latent compressed space.
- Compress s into a latent code, then count occurrences of the code.
- How do we get the image encoding? E.g, using autoencoders.

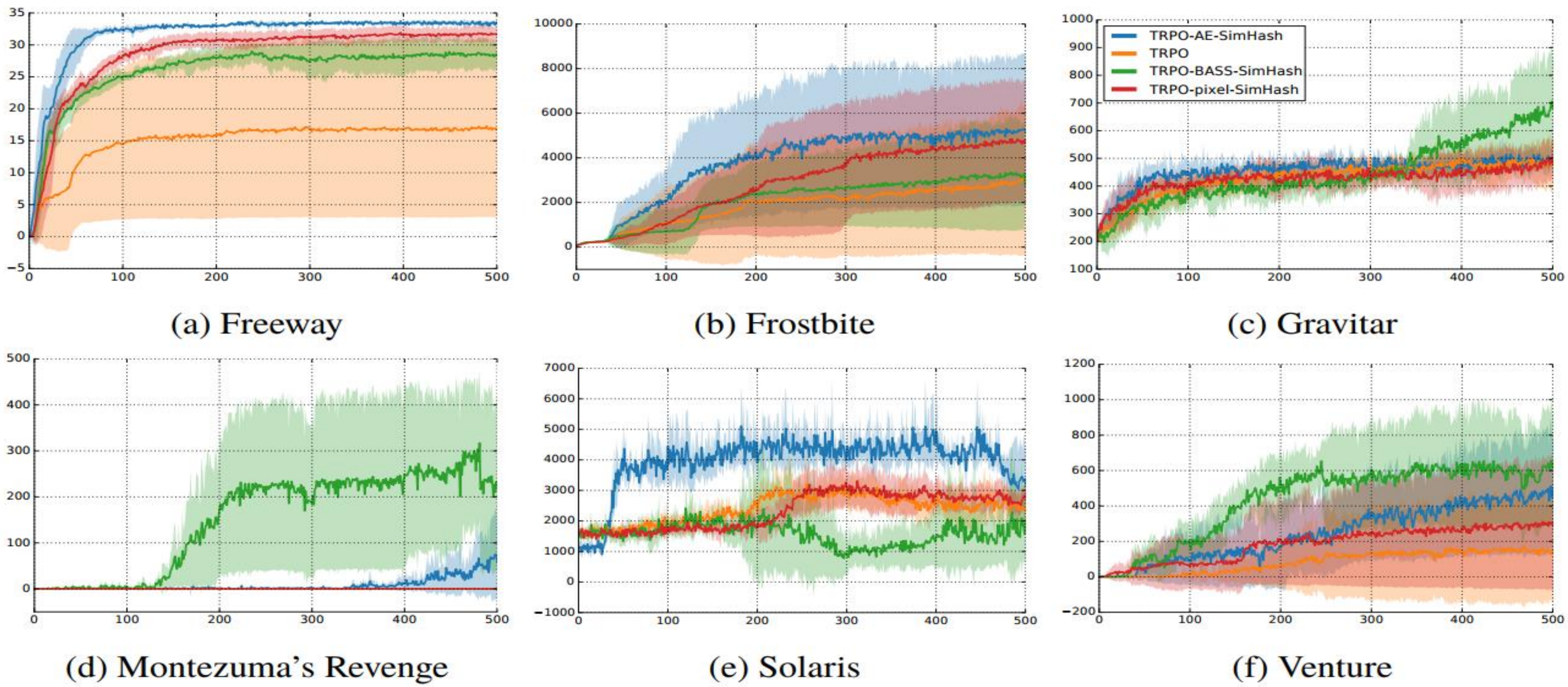


- Note: There is no guarantee such reconstruction loss will capture the important things that make two states to be similar or not policy wise..

#Exploration- A Study of Count-Based Exploration for Deep Reinforcement Learning, Tang et al.

State Counting with DeepHash

- We still count states (images) but not in pixel space, but in latent compressed space.
- Compress s into a latent code, then count occurrences of the code.
- How do we get the image encoding? E.g, using autoencoders.



(a) Freeway (b) Frostbite (c) Gravitar

(d) Montezuma's Revenge (e) Solaris (f) Venture

#Exploration- A Study of Count-Based Exploration for Deep Reinforcement Learning, Tang et al.

Curiosity-driven exploration

Seek **novelty/surprise**:

- Visit *novel states* s
- Observe *novel state transitions* (s,a)->s’

Mental models

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

-- Kenneth Craik, 1943, Chapter 5, page 61

That was what model based RL was all about.
Now we will be exploring so that our model improves the fastest!

[credit: Jitendra Malik]

Computational Curiosity

- “The direct goal of curiosity and boredom is to improve the **world model**. The indirect goal is to ease the learning of new goal-directed action sequences.”
- “The same complex mechanism which is used for ‘normal’ goal-directed learning is used for implementing curiosity and boredom. There is no need for devising a separate system which aims at improving the world model.”
- “Curiosity Unit”: reward is a function of the mismatch between model’s current predictions and actuality. There is positive reinforcement whenever the system fails to correctly predict the environment.
- “Thus the usual credit assignment process ... encourages certain past actions in order to repeat situations similar to the mismatch situation.” (planning to make your (internal) world model to fail)

Jurgen Schmidhuber, 1991, 1991, 1997

Computational Curiosity

In other words:

- Model learning and model improvement can be cast as the goals of goal-seeking behaviour.
- My goal is not to beat Atari but to improve my Atari model.
- OK. What is my reward then that trying to maximize that reward will lead to fast model learning?

Reward Prediction Error

Add exploration reward bonuses that encourage policies to visit states that will cause the prediction model to fail.

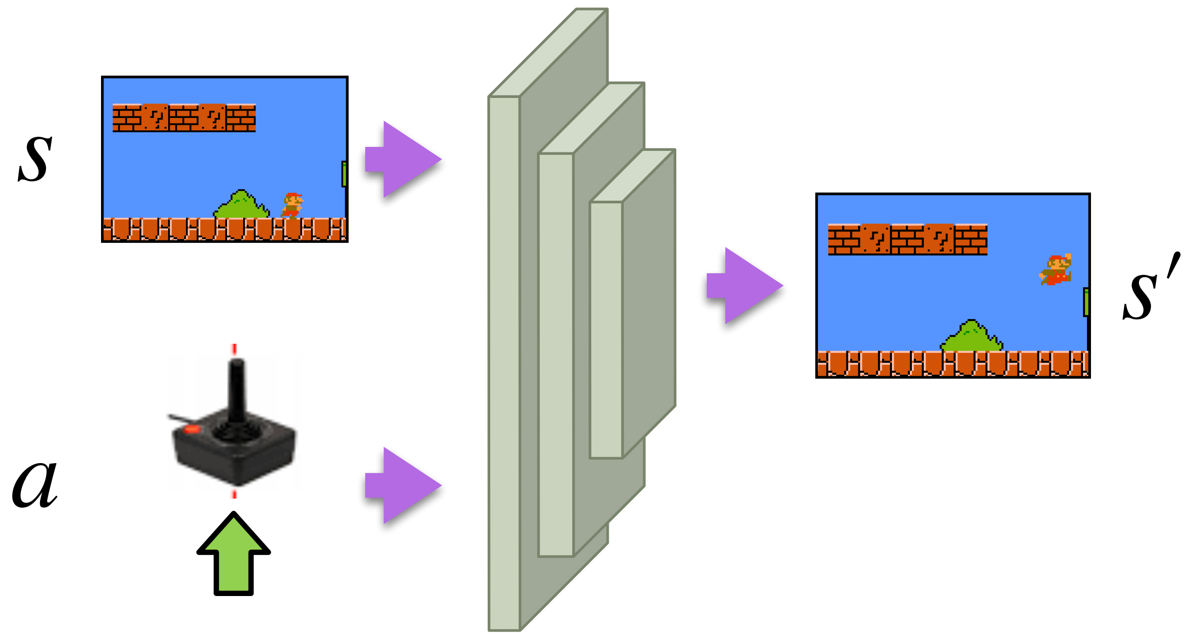
model error!

$$R^t(s, a, s') = \underbrace{r(s, a, s')}_{\text{extrinsic}} + \underbrace{\mathcal{B}^t(\|T(s, a; \theta) - s'\|)}_{\text{intrinsic}}$$

Note: we will be using T(s,a;\theta) to denote the dynamics (transition) function.

Learning Visual Dynamics

Exploration reward bonus $\mathscr{B}^t(s, a, s') = \|\mathrm{T}(s, a; \theta) - s'\|$



$$\min_{\theta} . \quad \|\mathrm{T}(s, a; \theta) - s'\|$$

Here we predict the visual observation!

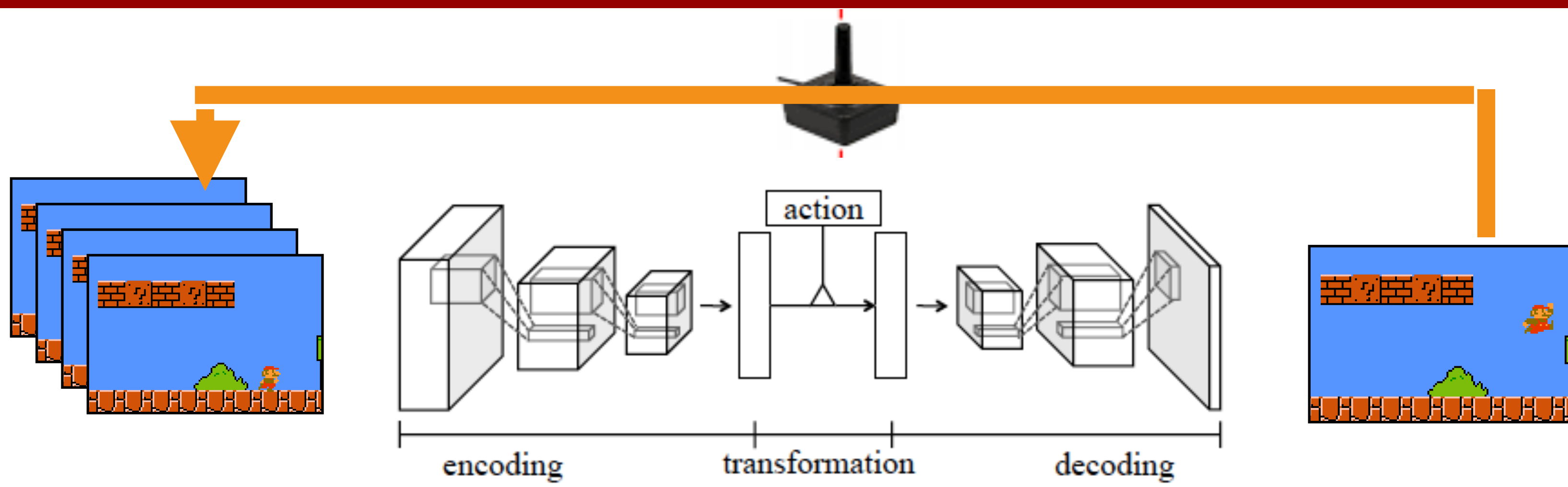
$$R^t(s, a, s') = \underbrace{r(s, a, s')}_{\text{extrinsic}} + \underbrace{\mathscr{B}^t(s, a, s')}_{\text{intrinsic}}$$

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
- Use this frame prediction to come up with an exploratory behavior in DQN: choose the action that leads to frames that are most dissimilar to a buffer of recent frames

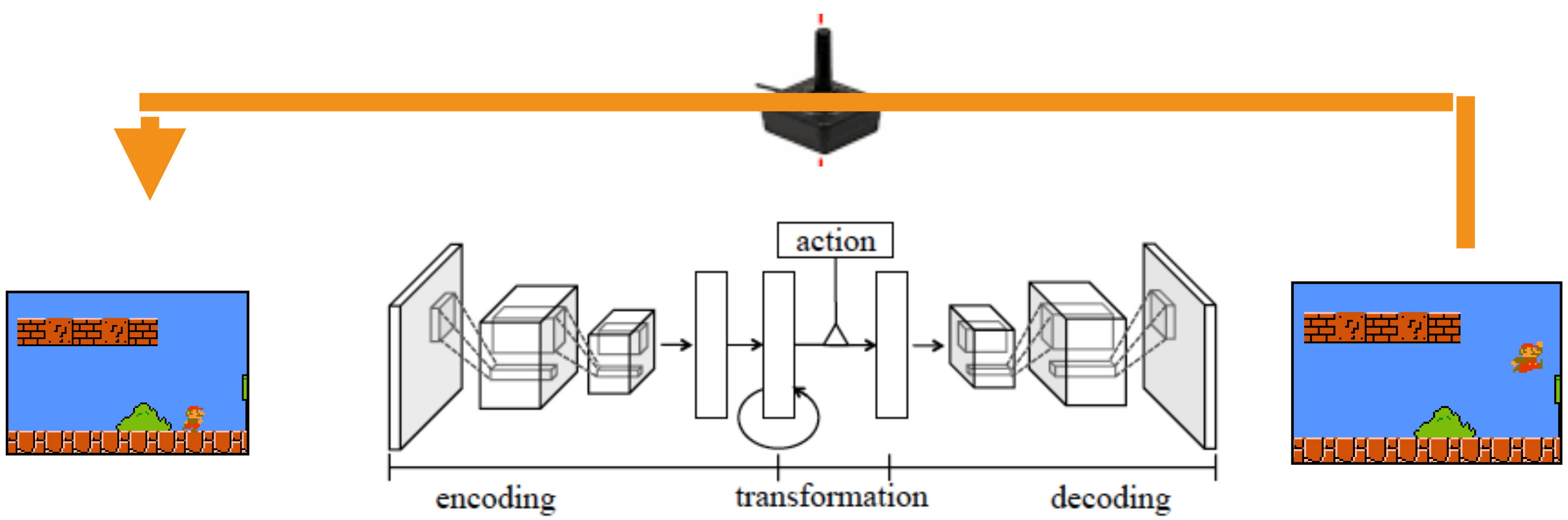
Frame prediction



(a) Feedforward encoding

Multiplicative interactions between action and hidden state (not concatenation):

$$h^{dec}_{t,i} = \sum_{j,l} W_{ijl} h^{enc}_{t,j} a_{t,l} + b_i,$$



(b) Recurrent encoding

Unroll the model by feeding the prediction back as input!

Progressively increase k (the length of the conditioning history) so that we do not feed garbage predictions as input to the predictive model:

$$\mathcal{L}_K(\theta) = \frac{1}{2K} \sum_i \sum_t \sum_{k=1}^{K} \left\| \hat{\mathbf{x}}^{(i)}_{t+k} - \mathbf{x}^{(i)}_{t+k} \right\|^2$$

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

Small objects are missed, e.g., the bullets. It is because they induce a tiny mean pixel prediction loss (despite the fact they may be task-relevant)

Frame prediction for Exploration

Algorithm 1 Deep Q-learning with informed exploration

Allocate capacity of replay memory R
Allocate capacity of trajectory memory D
Initialize parameters θ of DQN
while $steps < M$ **do**
  Reset game and observe image x_1
  Store image x_1 **in** D
  for t=1 to T **do**
    Sample c from Bernoulli distribution with parameter ϵ
    Set $a_t = \begin{cases} \mathbf{argmin}_a \, n_D\left(x_t^{(a)}\right) & \text{if } c = 1 \\ \operatorname{argmax}_a Q\left(\phi\left(s_t\right), a; \theta\right) & \text{otherwise} \end{cases}$
    Choose action a_t, observe reward r_t and image x_{t+1}
    Set $s_{t+1} = x_{t-2:t+1}$ and preprocess images $\phi_{t+1} = \phi\left(s_{t+1}\right)$
    Store image x_{t+1} **in** D
    Store transition $\left(\phi_t, a_t, r_t, \phi_{t+1}\right)$ in R
    Sample a mini-batch of transitions $\{\phi_j, a_j, r_j, \phi_{j+1}\}$ from R
    Update θ based on the mini-batch and Bellman equation
    $steps = steps + 1$
  end for
end while

Minimize similarity to a trajectory memory

Model	Seaquest	S. Invaders	Freeway	QBert	Ms Pacman
DQN - Random exploration	13119 (538)	698 (20)	30.9 (0.2)	3876 (106)	2281 (53)
DQN - Informed exploration	13265 (577)	681 (23)	32.2 (0.2)	8238 (498)	2522 (57)

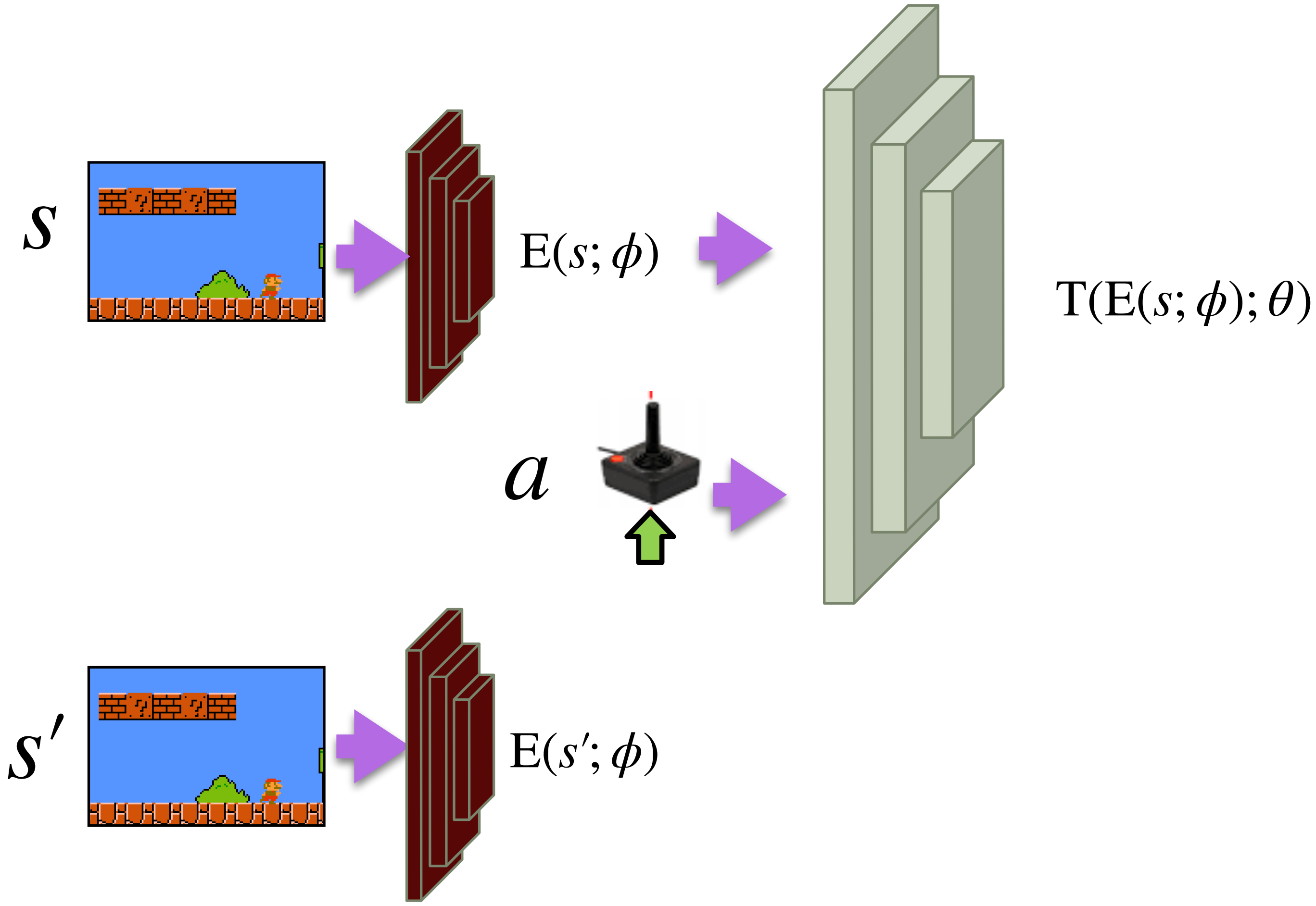
Predicting Raw Sensory Input (Pixels)

Should our prediction model be predicting the input observations?

- Observation prediction is difficult especially for high dimensional observations.
- Observation contains a lot of information unnecessary for planning, e.g., dynamically changing backgrounds that the agent cannot control and/or are irrelevant to the reward.

Learning Visual Dynamics

Exploration reward bonus $\mathscr{B}^t(s, a, s') = \|T(E(s; \phi), a; \theta) - E(s'; \phi)\|$



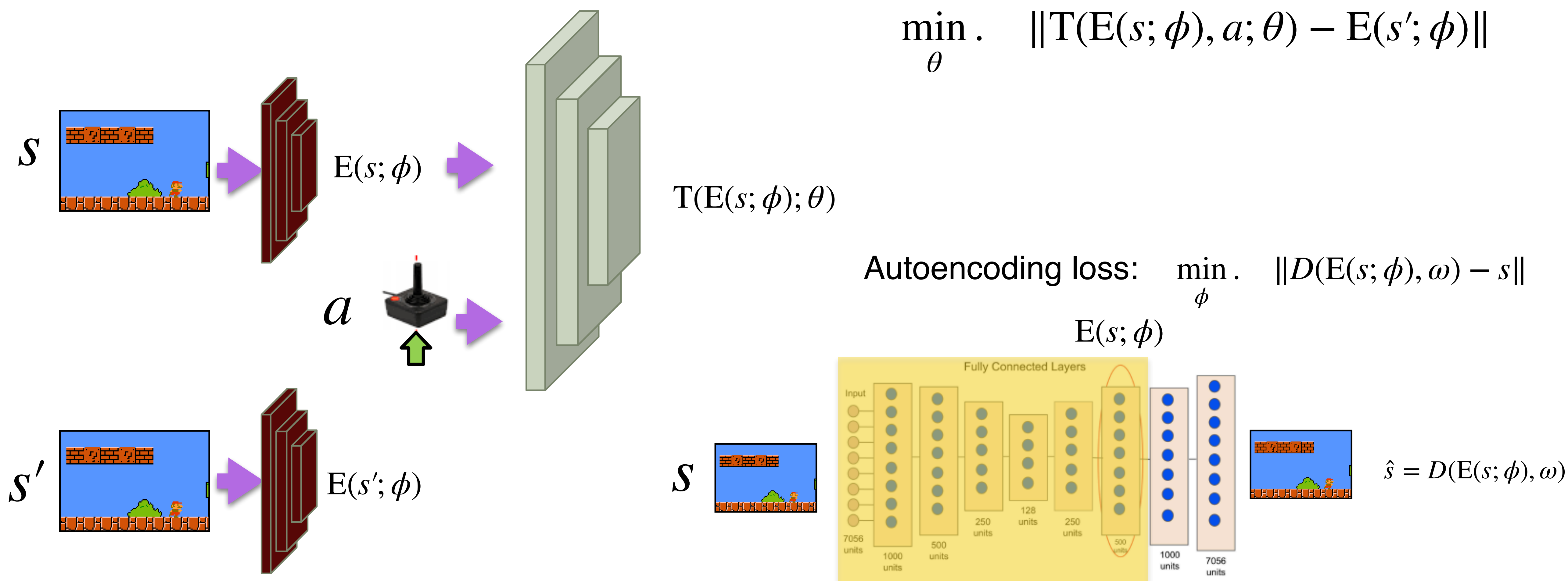
$$\min_{\theta,\phi} . \quad \|T(E(s; \phi), a; \theta) - E(s'; \phi)\|$$

What is the problem with this optimization problem?

There is a trivial solution :-(

Learning Visual Dynamics

Exploration reward bonus $\mathscr{B}^t(s, a, s') = \|T(E(s;\phi), a;\theta) - E(s';\phi)\|$



- Let's learn image encoding using autoencoders (to avoid the trivial solution)
- …and suffer the problems of autoencoding reconstruction loss that has little to do with our task

Incentivizing exploration in RL with deep predictive models, Stadie et al.

Explore guided by Novelty of Transition Dynamics

It uses the autoencoder solution!

Algorithm 1 Reinforcement learning with model prediction exploration bonuses

Initialize $\max_e = 1$, EpochLength, β, C
for iteration t in T **do**
  Observe $(s_t, a_t, s_{t+1}, \mathcal{R}(s_t, a_t))$
  Encode the observations to obtain $\sigma(s_t)$ and $\sigma(s_{t+1})$
  Compute $e(s_t, a_t) = \|\sigma(s_{t+1}) - \mathcal{M}_\phi(\sigma(s_t), a_t)\|_2^2$ and $\bar{e}(s_t, a_t) = \frac{e(s_t, a_t)}{\max_e}$.
  Compute $\mathcal{R}_{Bonus}(s_t, a_t) = \mathcal{R}(s, a) + \beta \left(\frac{\bar{e}_t(s_t, a_t)}{t * C} \right)$
  if $e(s_t, a_t) > \max_e$ **then**
    $\max_e = e(s_t, a_t)$
  end if
  Store $(s_t, a_t, \mathcal{R}_{bonus})$ in a memory bank Ω.
  Pass Ω to the reinforcement learning algorithm to update π.
  if $t \mod \text{EpochLength} == 0$ **then**
    Use Ω to update $\mathcal{M}$.
    Optionally, update σ.
  end if
end for
return optimized policy π

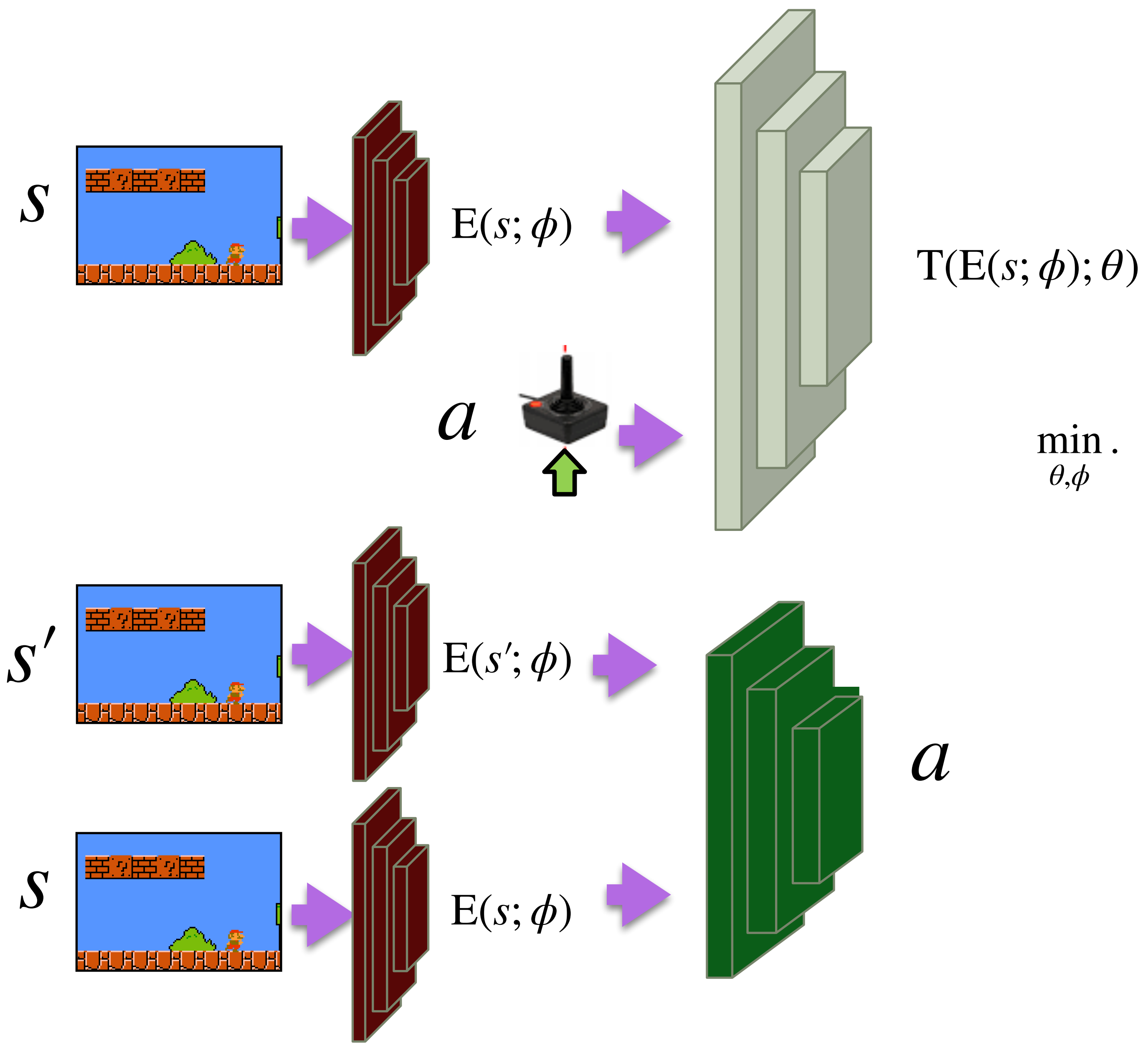
Such reward normalization is very important! Because exploration rewards during training are non-stationary, such scale normalization helps accelerate learning.

The autoencoder is trained as data arrives

Incentivizing exploration in RL with deep predictive models, Stadie et al.

Learning Visual Dynamics

Exploration reward bonus $\mathscr{B}^t(s, a, s') = \|\mathrm{T}(\mathrm{E}(s;\phi), a;\theta) - \mathrm{E}(s';\phi)\|$



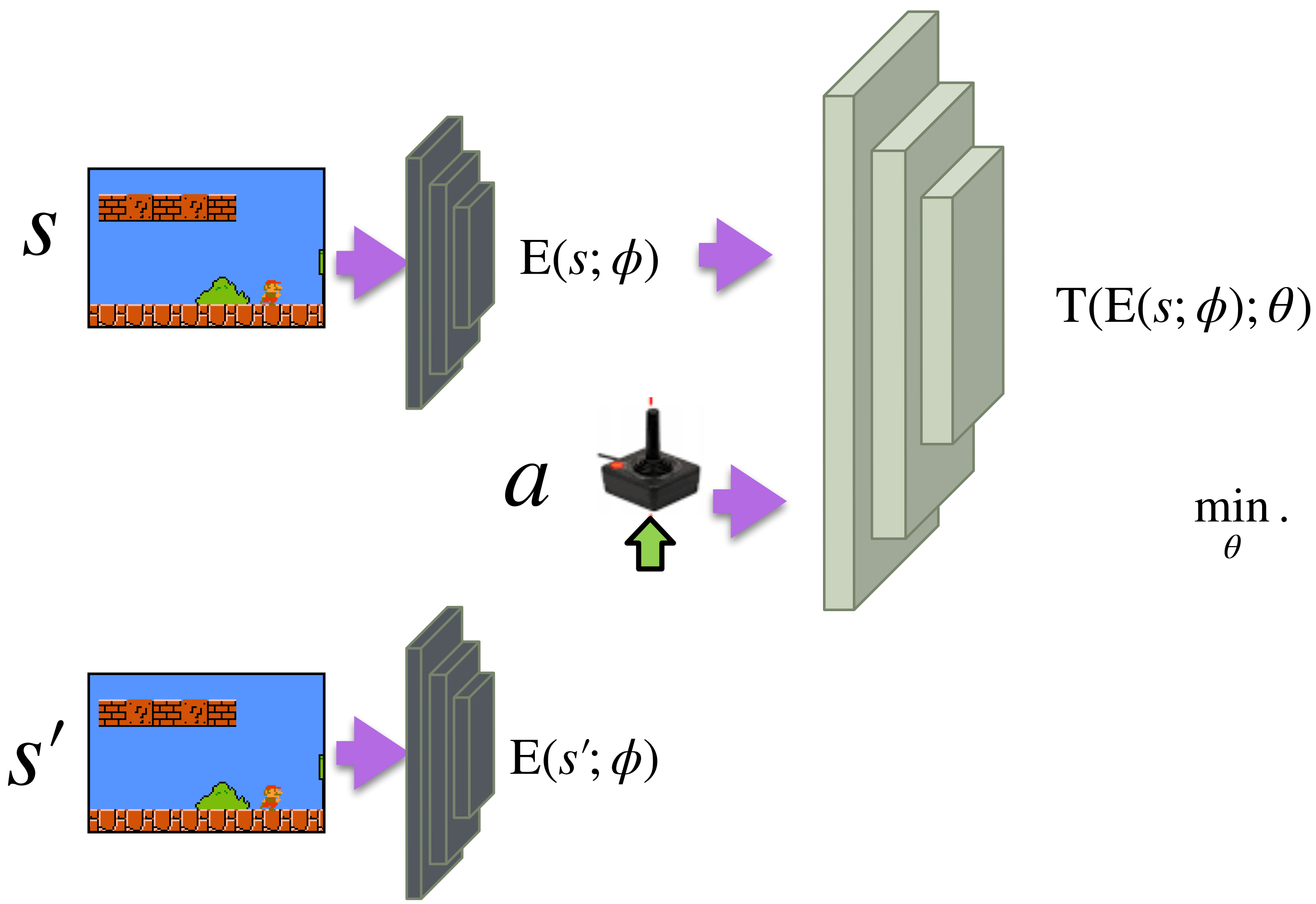
$$\min_{\theta,\phi} . \quad \|\mathrm{T}(\mathrm{E}(s;\phi), a;\theta) - \mathrm{E}(s';\phi)\| + \|\mathrm{Inv}(\mathrm{E}(s;\phi), \mathrm{E}(s;\phi);\psi) - a\|$$

- Let's couple forward and inverse models (to avoid the trivial solution)
- …then we will only predict things that the agent can control

Curiosity driven exploration with self-supervised prediction, Pathak et al.

Learning Visual Dynamics

Exploration reward bonus $\mathscr{B}^t(s, a, s') = \|\mathrm{T}(\mathrm{E}(s;\phi), a;\theta) - \mathrm{E}(s';\phi)\|$



$$\min_{\theta}. \quad \|\mathrm{T}(\mathrm{E}(s;\phi), a;\theta) - \mathrm{E}(s';\phi)\| + \|\mathrm{Inv}(\mathrm{E}(s;\phi), \mathrm{E}(s;\phi);\theta) - a\|$$

- Let's use random neural networks (networks initialized randomly and frozen thereafter)
- …and be embarassed about how well it works on Atari games

Large-scale study of Curiosity-Driven Learning, Burda et al.

Task Versus Exploration rewards

Exploration reward bonus $\mathscr{B}^t(s,a,s') = \|\mathrm{T}(\mathrm{E}(s;\phi),a;\theta) - \mathrm{E}(s';\phi)\|$

Only task reward: $R(s,a,s') = \underbrace{r(s,a,s')}_{\text{extrinsic}}$

Task+curiosity: $R^t(s,a,s') = \underbrace{r(s,a,s')}_{\text{extrinsic}} + \underbrace{\mathscr{B}^t(s,a,s')}_{\text{intrinsic}}$

Sparse task + curiosity: $R^t(s,a,s') = \underbrace{r^T(s,a,s')}_{\text{extrinsic terminal}} + \underbrace{\mathscr{B}^t(s,a,s')}_{\text{intrinsic}}$

Task Versus Exploration rewards

Exploration reward bonus $\mathscr{B}^t(s, a, s') = \|\mathrm{T}(\mathrm{E}(s; \phi), a; \theta) - \mathrm{E}(s'; \phi)\|$

Only task reward: $R(s, a, s') = \underbrace{r(s, a, s')}_{\text{extrinsic}}$

Task+curiosity: $R^t(s, a, s') = \underbrace{r(s, a, s')}_{\text{extrinsic}} + \underbrace{\mathscr{B}^t(s, a, s')}_{\text{intrinsic}}$

Sparse task + curiosity: $R^t(s, a, s') = \underbrace{r^T(s, a, s')}_{\text{extrinsic terminal}} + \underbrace{\mathscr{B}^t(s, a, s')}_{\text{intrinsic}}$

Only curiosity: $R^t(s, a, s') = \underbrace{\mathscr{B}^t(s, a, s')}_{\text{intrinsic}}$

- Train an A3C agent under only curiosity reward.
- Will it learn to do something useful?

Curiosity driven exploration with self-supervised prediction, Pathak et al
Large-scale study of Curiosity-Driven Learning, Burda et al

Policy Transfer

Policies trained with A3C using only curiosity rewards
Prediction error using forward/inverse model coupling

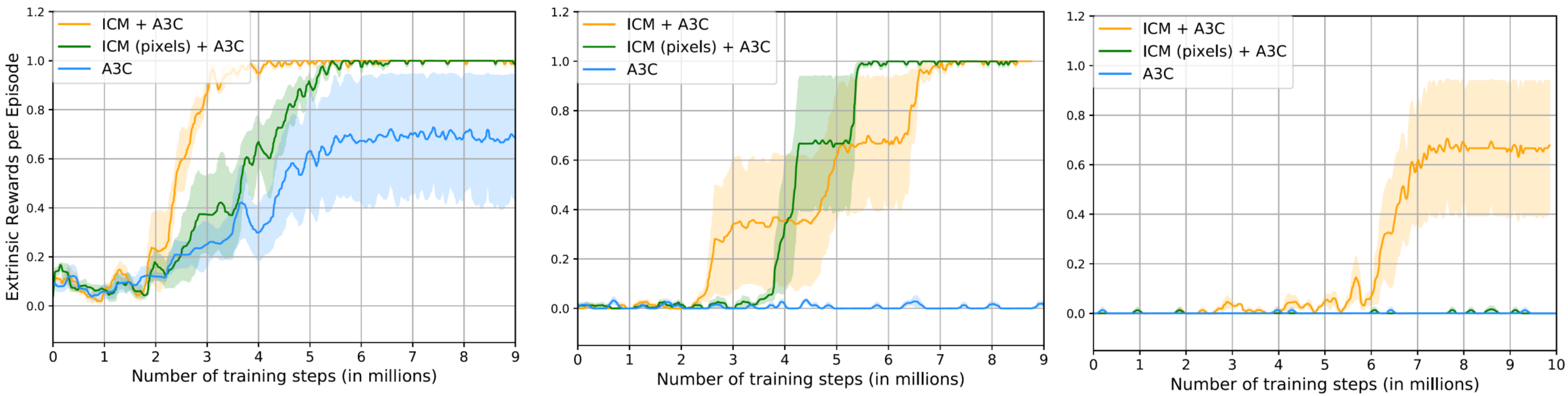
Trained on Level-1

Testing on Level-2

Large-scale study of Curiosity-Driven Learning, Burda et al.

Curiosity helps even more when rewards are sparse

A3C: extrinsic only reward
ICM+A3C: exploration+extrinsic, where model learning happens in learned feature space
ICM(pixel)+A3C: exploration+extrinsic, where model learning happens in pixel space



(a) "dense reward" setting (b) "sparse reward" setting (c) "very sparse reward" setting

Conclusions

- Using curiosity as a reward results in policies that collect much higher task rewards than policies trained under task reward alone - so curiosity (as prediction error) a good proxy for task rewards

Curiosity driven exploration with self-supervised prediction, Pathak et al.

No(extrinsic)rewardRL is not new

- Itti, L., Baldi, P.F.: Bayesian surprise attracts human attention. In: NIPS'05. pp. 547–554 (2006)
- Schmidhuber, J.: Curious model-building control systems. In: IJCNN'91. vol. 2,pp. 1458–1463 (1991)
- Schmidhuber, J.: Formal theory of creativity, fun, and intrinsic motivation (1990-2010). Autonomous Mental Development, IEEE Trans. on Autonomous MentalDevelopment 2(3), 230–247 (9 2010)
- Singh, S., Barto, A., Chentanez, N.: Intrinsically motivated reinforcement learning.In: NIPS'04 (2004)
- Storck, J., Hochreiter, S., Schmidhuber, J.: Reinforcement driven information acquisition in non-deterministic environments. In: ICANN'95 (1995)
- Sun, Y., Gomez, F.J., Schmidhuber, J.: Planning to be surprised: Optimal bayesian exploration in dynamic environments (2011), http://arxiv.org/abs/1103.5708

Limitation of Prediction Error as Bonus

- Agent will be rewarded even though the model cannot improve.
- The agent is attracted forever in the most noisy states, with unpredictable outcomes.
- If we give the agent a TV and a remote, it becomes *a couch potato*!

Large-scale study of Curiosity-Driven Learning, Burda et al.