Reinforcement learning
Episode -1, part -1

Hierarchical RL 101

Slides by Pavel Shvechikov

Yandex Data Factory

LAMBDA

British Hedgehog Preservation Society

RL agents vs humans:

- Model-free: trial & error (mostly error)
- Model-based: planning & execution

RL agents vs humans:

- Model-free: trial & error (mostly error)
- Model-based: planning & execution
- Action values: understanding consequences of actions
- Exploration: curiosity

RL agents vs humans:

- Model-free: trial & error (mostly error)
- Model-based: planning & execution
- Action values: understanding consequences of actions
- Exploration: curiosity
- **???** learning to remember past events
- **???** understanding others' motives
- **???** learning by imitation (cpt. obvious)
- **???** accounting for others' interests

RL agents vs humans:

- Model-free: trial & error (mostly error)
- Model-based: planning & execution
- Action values: understanding consequences of actions
- Exploration: curiosity
- POMDP: learning to remember past events
- Inverse RL: understanding others’ motives
- Imitation learning: learning by imitation (cpt. obvious)
- Multi-agent RL: accounting for others’ interests

How humans work:

- Don't seem to follow epsilon-greedy exploration policy (see exploration lecture)

- Think in several layers of abstraction
  - “Contract leg muscles”
  - “Push gas pedal (while driving)”
  - “Take left turn in 15 meters”
  - “Drive to school”,
  - “Give my children education”

Problem: Rewards are usually sparse (rare) and/or delayed in time.

It takes exponentially more random exploration to learn optimal policy in case of rare rewards.

Humans:

- Don't seem to follow epsilon-greedy exploration policy
  (see exploration lecture)

- Think in several layers of abstraction
  - “Contract leg muscles”
  - “Push gas pedal (while driving)”
  - “Take left turn in 15 meters”
  - “Drive to school”,
  - “Give my children education”

We want hierarchical decision-making!

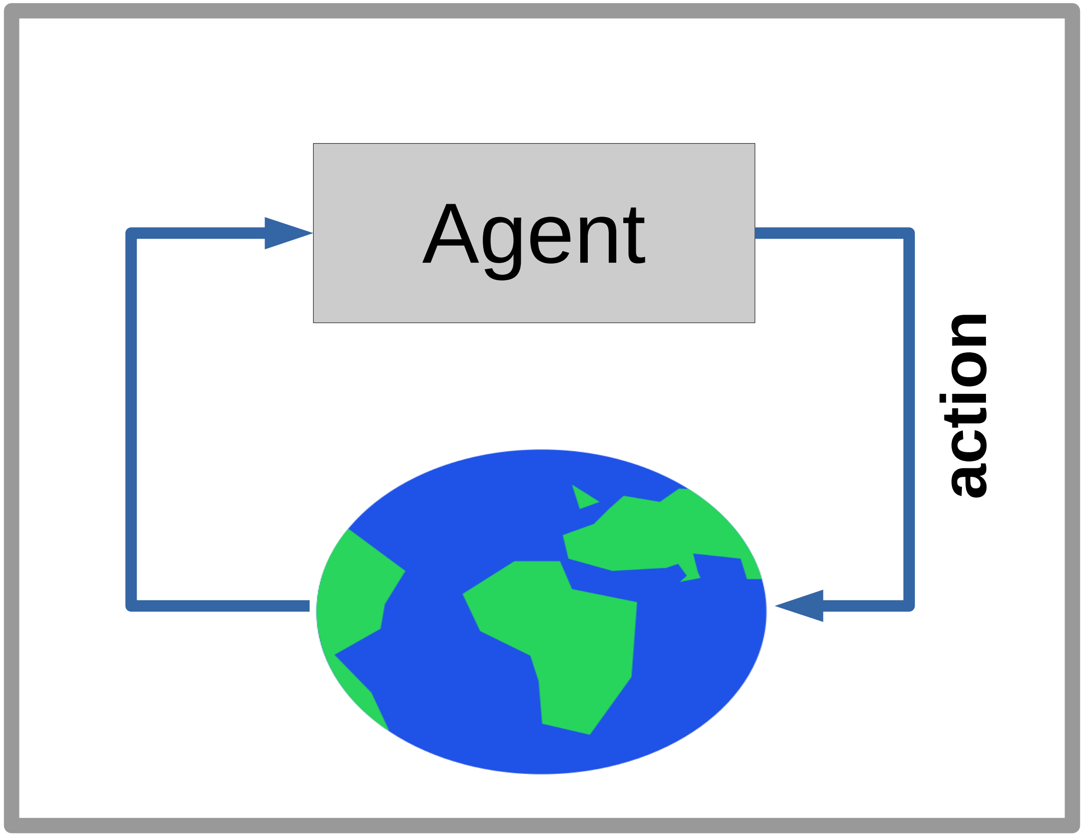
So what is hierarchy, again?

Suggestions?

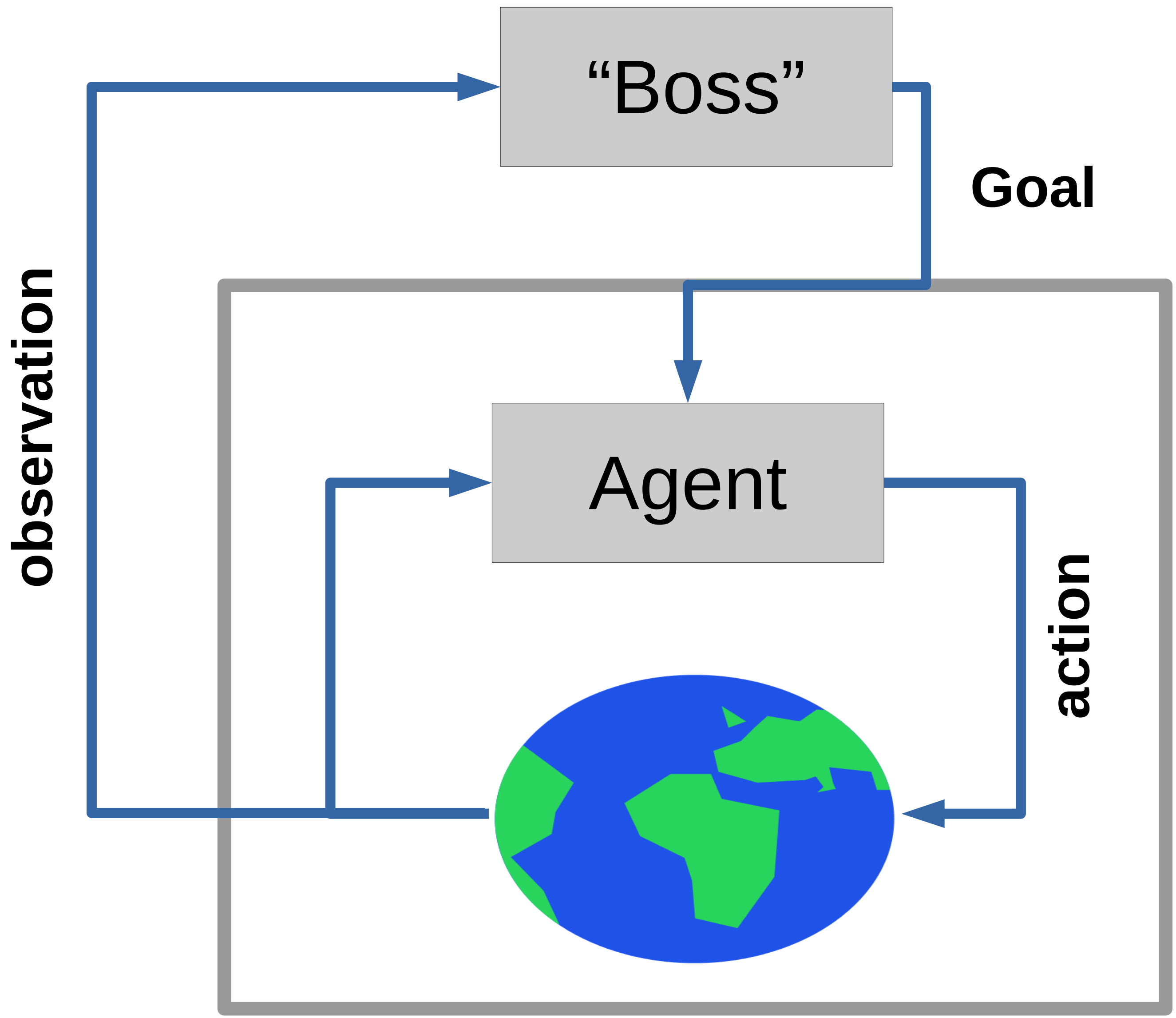
So what is hierarchy, again?

- I know, I know! It's about operating in term of more abstract states!
- No! It's about acting on a longer time scale!
- No! it's about decomposing reward into short-term and long-term

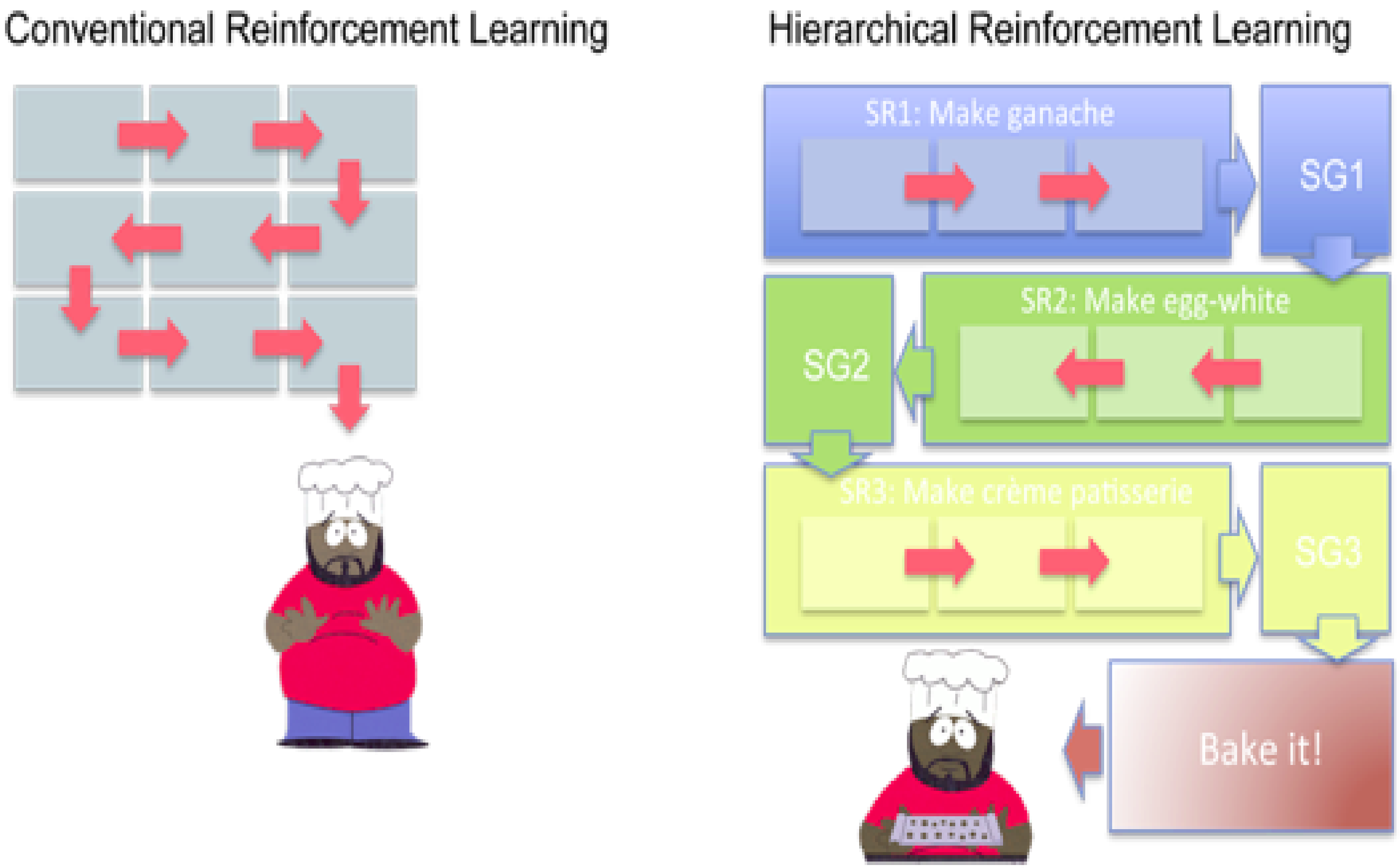
Normal MDP



Hierarchical MDP

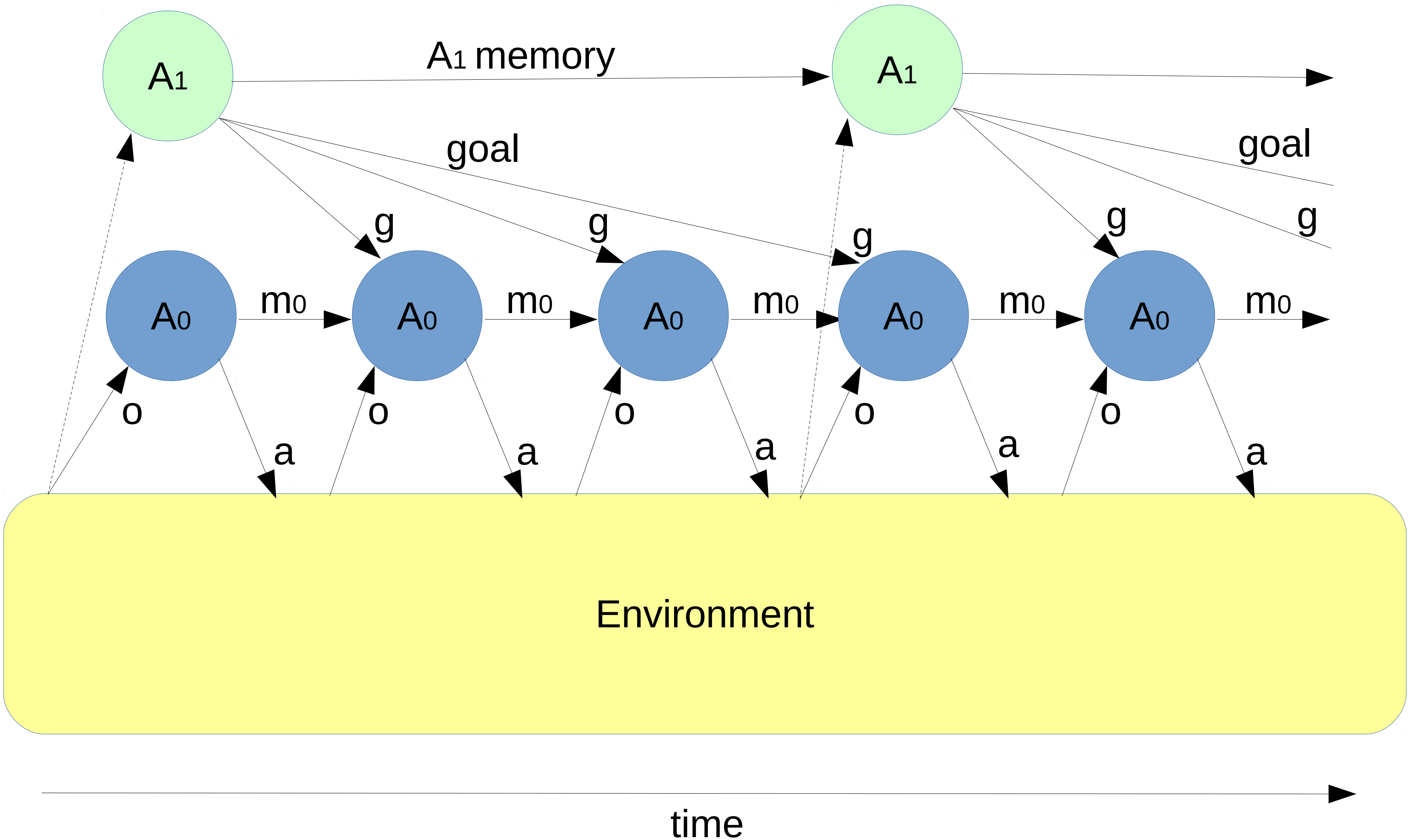


Options



The “option" method by Sutton, Precup, and Singh 1998

Naive Hierarchical RL



Naive Hierarchical RL

Model-based

- Manual measurable goals
- e.g. arxiv: 1604.06057

Model-free

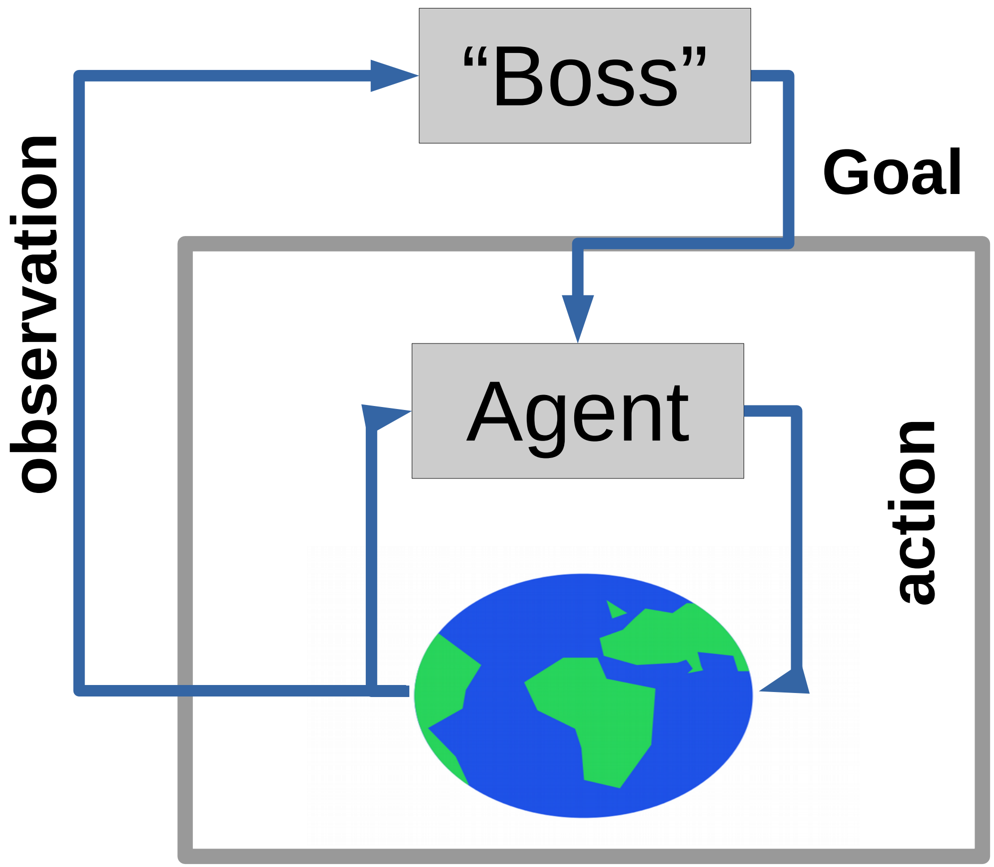
- Agent needs to “learn” useful goals
- Naive: no better than standard rl

FeUdal Networks

Sketch:

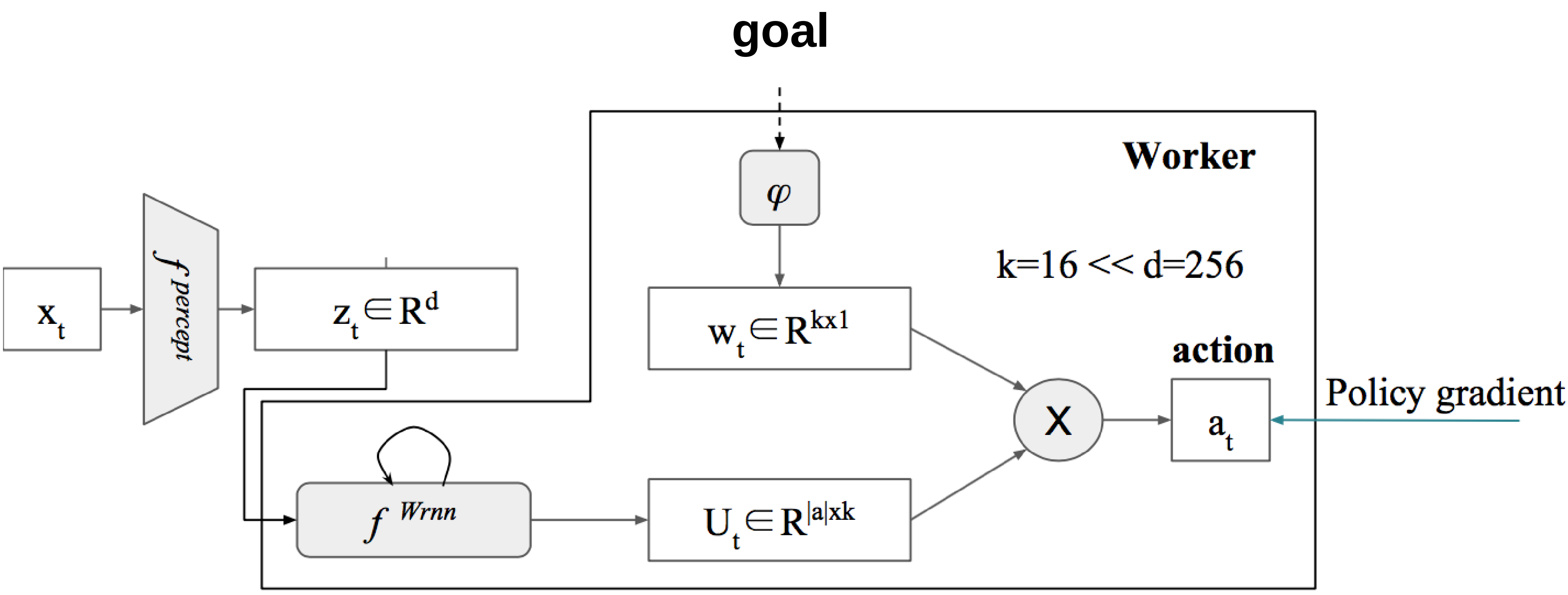
Goal = direction in state space

Differentiable End-to-end

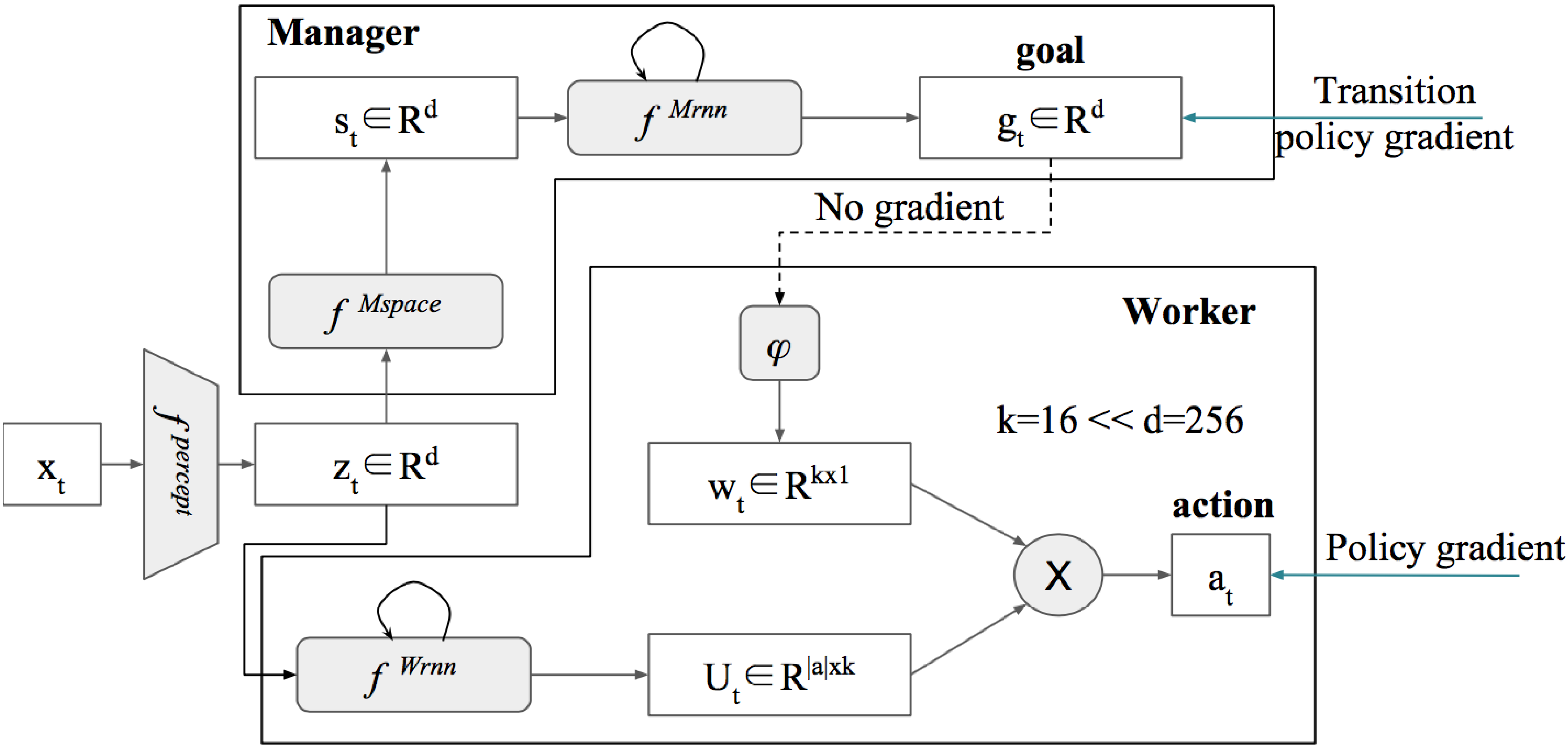


Arxiv: 1703.01161

FeUdal Networks



FeUdal Networks



Common

$z_t = f^{percept}(x_t)$ is a prepossessed observation x_t

Manager

$$s_t = f^{Mspace}(z_t)$$
$$(h_t^M, \widehat{g}_t) = f^{Mrnn}(s_t, h_{t-1}^M)$$
$$g_t = \widehat{g}_t / ||\widehat{g}_t||$$

s_t – latent state representation

g_t – goal vector

h_t^M – Manager rnn hidden state

Common

$z_t = f^{percept}(x_t)$ is a prepossessed observation x_t

Manager

$$s_t = f^{Mspace}(z_t)$$

$$(h_t^M, \hat{g}_t) = f^{Mrnn}(s_t, h_{t-1}^M)$$

$$g_t = \hat{g}_t / ||\hat{g}_t||$$

Worker

$$w_t = \varphi\left(\sum_{i=t-c}^{t} g_i\right)$$

$$h_t^W, U_t = f^{Wrnn}(z_t, h_{t-1}^W)$$

$$\pi_t = \mathrm{Softmax}(U_t w_t)$$

s_t – latent state representation

g_t – goal vector

h_t^M – Manager rnn hidden state

w_t – goal embedding

$\varphi(\cdot)$ – linear, **without bias**

U_t – matrix, output of rnn

h_t^W – Worker rnn hidden state

Update rules: manager

Boss' objective

$$\nabla g_t = \left(R_t - V_t^M(x_t, \theta)\right) \cdot \nabla d_{\cos}\left(s_{t+c} - s_t, g_t(\theta)\right)$$

- $\left(s_{t+c} - s_t\right)$ - agent's path in state space (embedding)
- $d_{\cos}\left(s_{t+c} - s_t, g_t(\theta)\right)$ - cosine similarity
- $\left(R_t - V_t^M(x_t, \theta)\right)$ - advantage

Q: Reminds you of something?

Update rules: manager

Boss' objective

$$\nabla g_t = \left(R_t - V_t^M(x_t, \theta)\right) \cdot \nabla d_{\cos}\left(s_{t+c} - s_t, g_t(\theta)\right)$$

- $\left(s_{t+c} - s_t\right)$ - agent's path in state space (embedding)
- $d_{\cos}\left(s_{t+c} - s_t, g_t(\theta)\right)$ - cosine similarity
- $\left(R_t - V_t^M(x_t, \theta)\right)$ - advantage
- Policy gradient for von Mises-Fisher distribution

$$\pi_{boss}\left(s_{t+c} - s_t | g(\theta)\right) \sim e^{d_{\cos}\left(s_{t+c} - s_t, g(\theta)\right)}$$

Update rules: worker

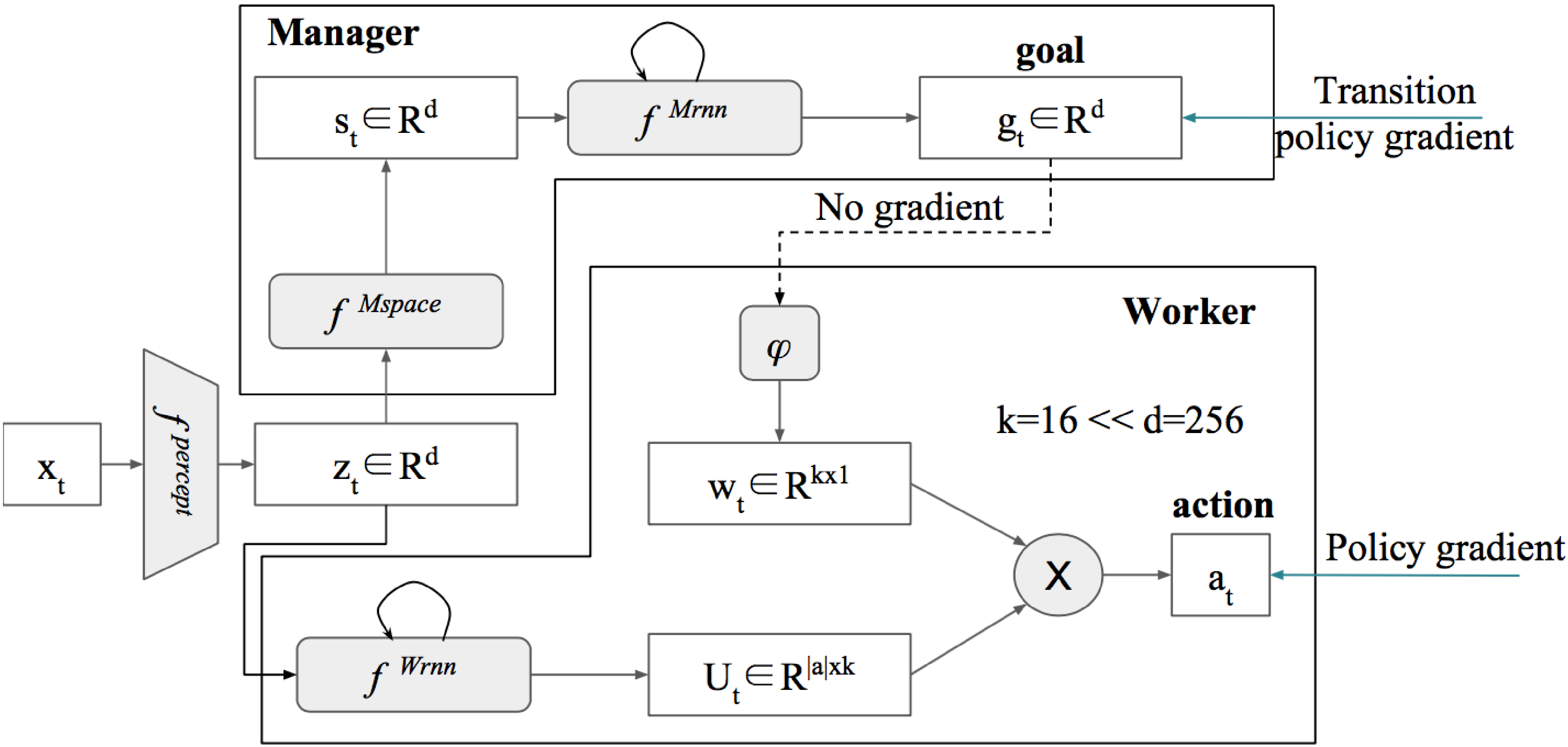
Intrinsic reward

$$\hat{R} = d_{\cos}\left(s_{t+c} - s_t, g_t(\theta)\right)$$

Worker updates:

$$\nabla J = E\left[d_{\cos}\left(s_{t+c} - s_t, g_t(\theta)\right) \cdot \nabla \log \pi_{worker}(a|s)\right]$$

That scheme again



More on FeUdal nets

More hacks:

- Dilated LSTM for manager (as Mrnn)

stack $\{\hat{h}\}_{i=1}^{r}$ of r = 10 states

$$\hat{h}_t^{t\%r}, g_t = LSTM\left(s_t, \hat{h}_{t-t}^{t\%r}\right)$$

- Only one group is updated per time-step
- Read more: arxiv.org/abs/1703.01161 , section 4.1
- Similar idea: arxiv.org/abs/1402.3511

More on FeUdal nets

More hacks:

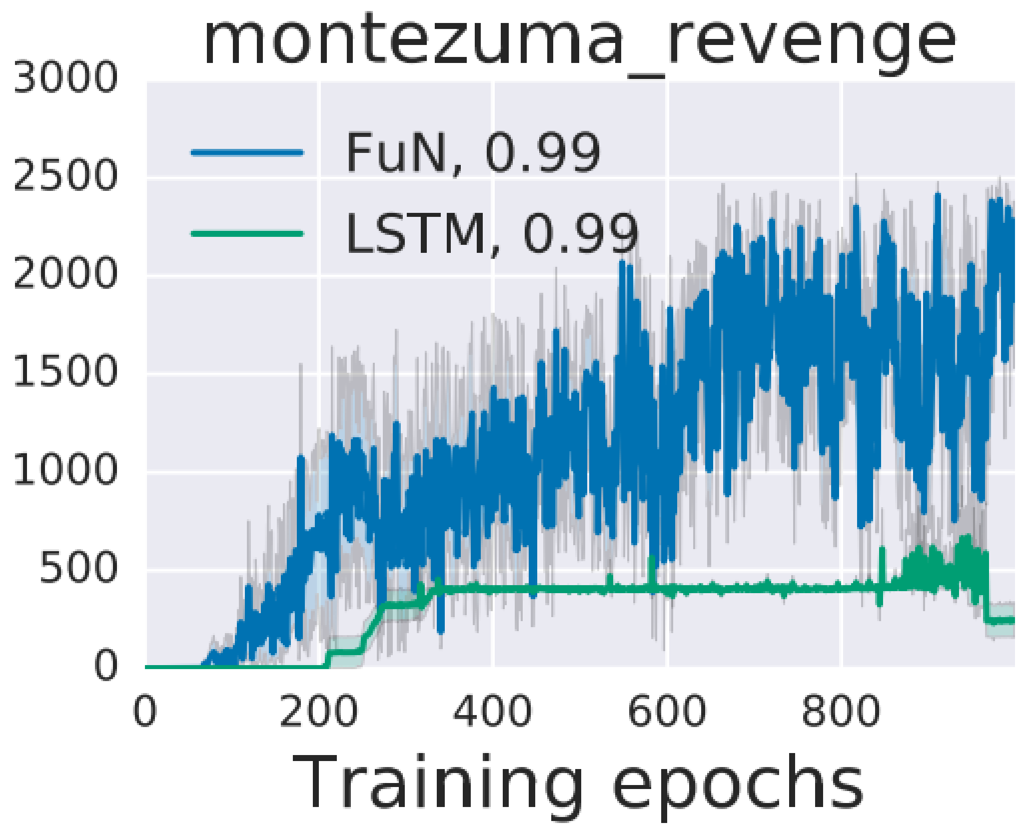
- Dilated LSTM for manager (as Mrnn)

stack $\{\hat{h}\}_{i=1}^{r}$ of r = 10 states

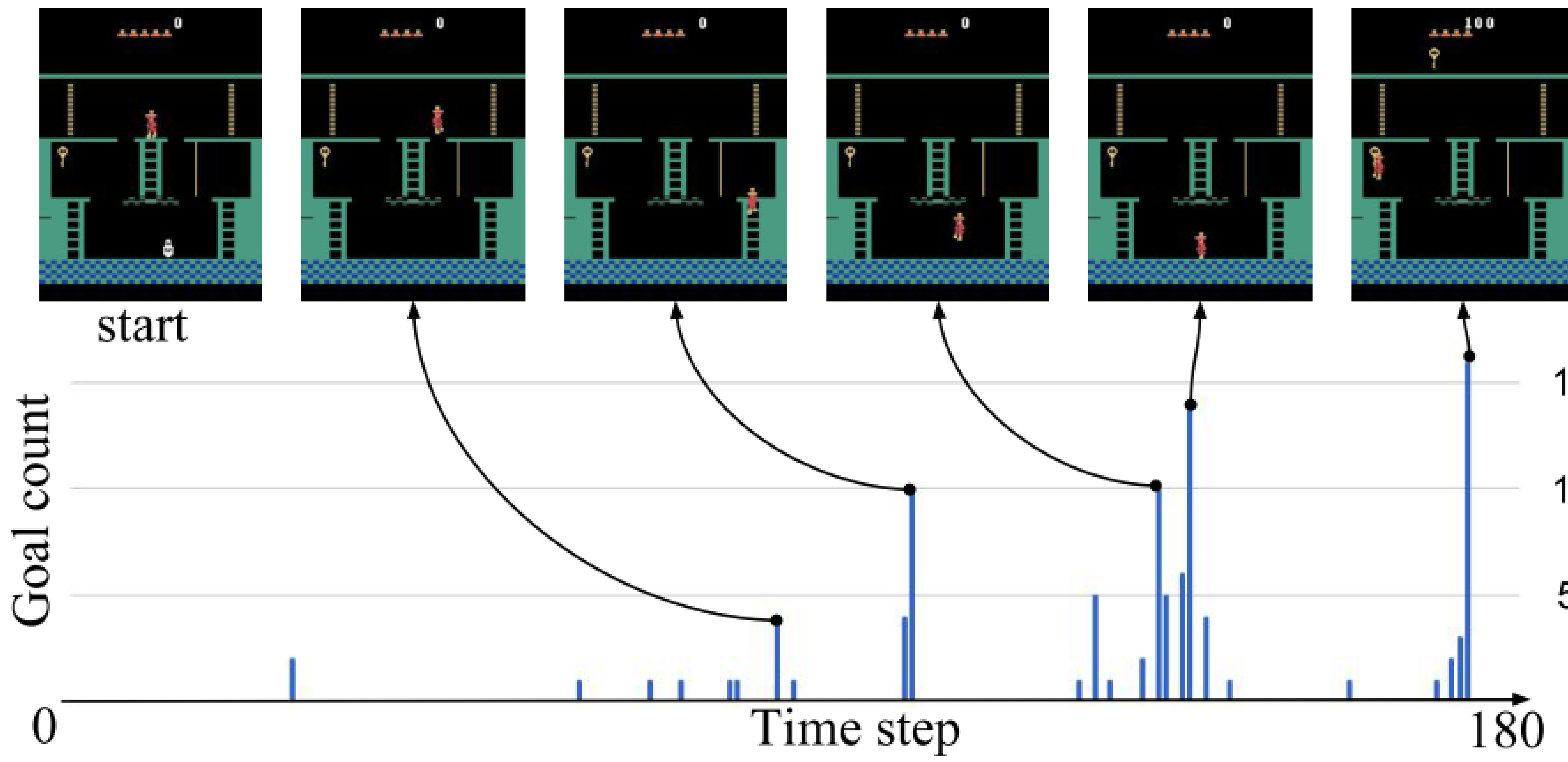
$$\hat{h}_t^{t\%r}, g_t = LSTM\left(s_t, \hat{h}_{t-t}^{t\%r}\right)$$

- Only one group is updated per time-step

Some results



Some results



More results

Course outro

the end...

Not too late to send homeworks.

Gonna be binge-checking on 19-20

Please tell us how to improve the course
http://bit.ly/2qwZSwN

Course outro

- Tons of bug-fixes
- Awesome research
- Useful feedback

Please tell us how to improve the course
http://bit.ly/2qwZSwN