FeUdal Networks for Hierarchical Reinforcement Learning

Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, Koray Kavukcuoglu

Topic: Hierarchical RL

Presenter: Théophile Gaudin

Why Hierarchical RL?

- RL is hard
  - Sparse reward
  - Long time-horizon

https://www.retrogames.cz/play_124-Atari2600.php?language=EN

- More “human-like” approach to decision making

Human-like decision making

When we type on a computer keyboard, we just thinking **about the words we want to write**. We don’t think about **each our fingers and muscles individually.**

We make hierarchical abstractions

Could this work for RL too?

Feudalism?

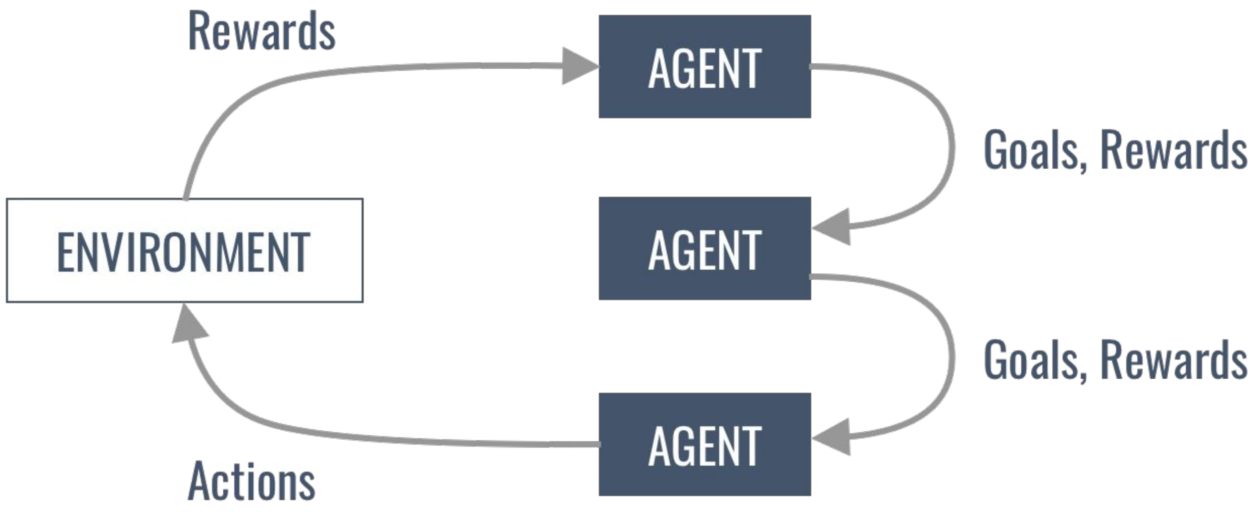
Governance system in Europe between 9-15th centuries

Top-down "management"

https://en.wikipedia.org/wiki/Feudalism

Feudal Reinforcement Learning (Dayan & Hinton 93')

- Only top Manager sees the environment reward
- Managers rewards and set goals for level below
- Managers are not aware of what happens at other level



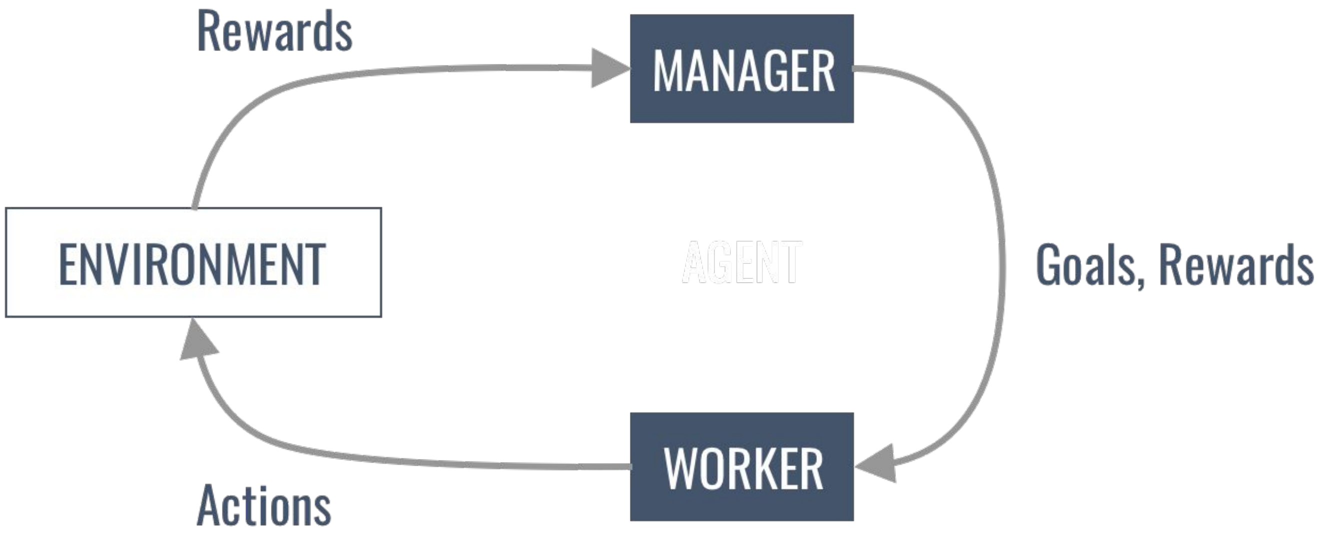
FeUdal Networks

Manager

- Lower temporal resolution
- Sets **directional** goals
- Rewarded by env.

Worker

- Higher temporal resolution
- Rewarded by the Manager
- Produces actions in env.



No gradient are propagated between the Manager and the Worker

Directional vs Absolute Goals

An absolute goal would be **to reach** a particular state

Ex: you have an address to reach

A direction goal would be **to go towards** a particular state

Ex: you have a direction to follow

Model Architecture Details

$$z_t = f^{\text{percept}}(x_t)$$

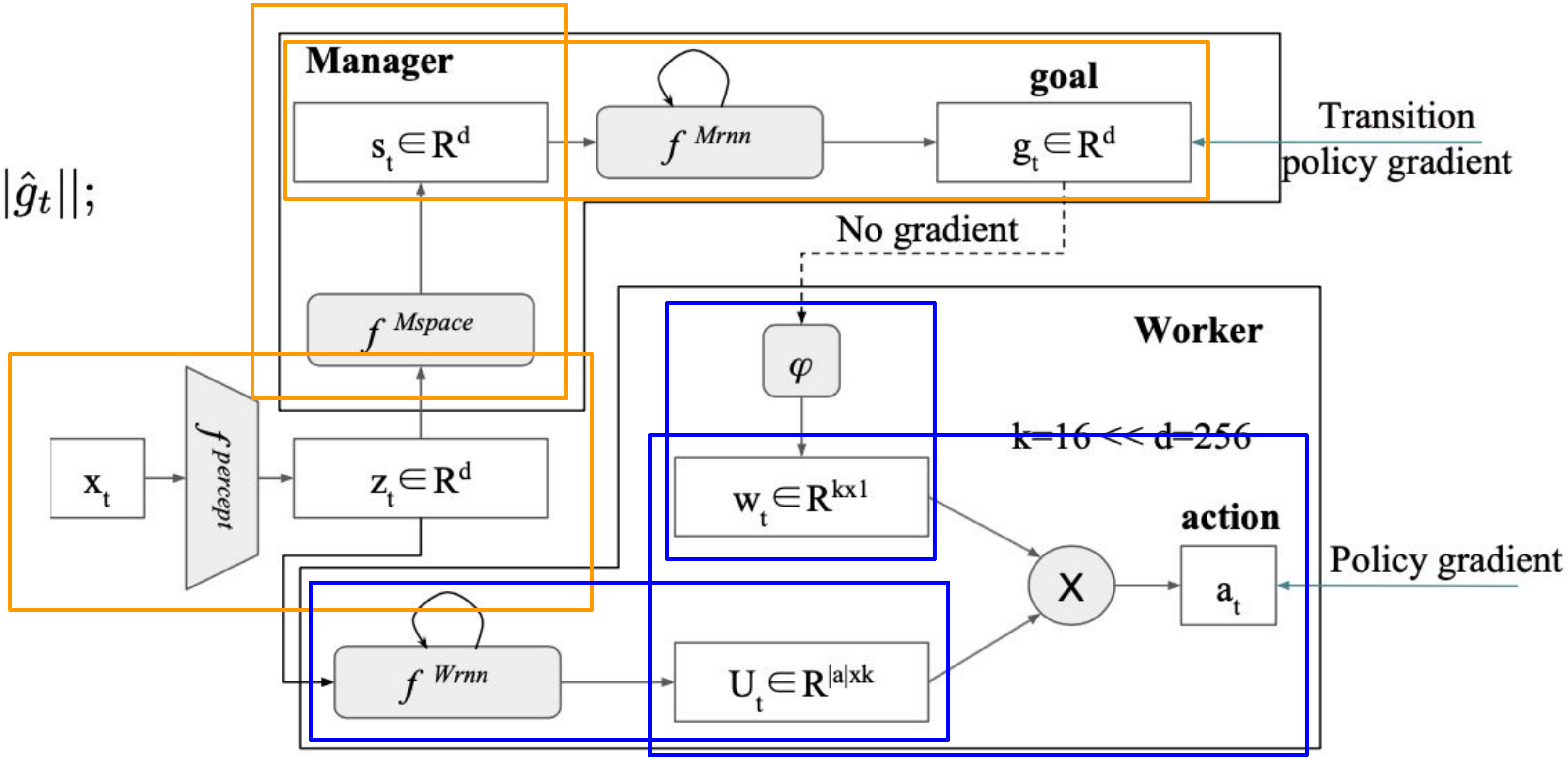
$$s_t = f^{Mspace}(z_t)$$

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

$$w_t = \phi(\sum_{i=t-c}^{t} g_i)$$

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

$$\pi_t = SoftMax(U_t w_t)$$



How to train this model?

- Could use TD-learning but then g_t would not have any semantic meaning
- Approximate transition policy gradient

Manager

$$\nabla g_t = A_t^M \nabla_\theta d_{\cos}(\underbrace{s_{t+c} - s_t}_{\text{Direction in the latent space}}, g_t(\theta)),$$

$$\text{where } A_t^M = R_t - V_t^M(x_t, \theta)$$

Worker

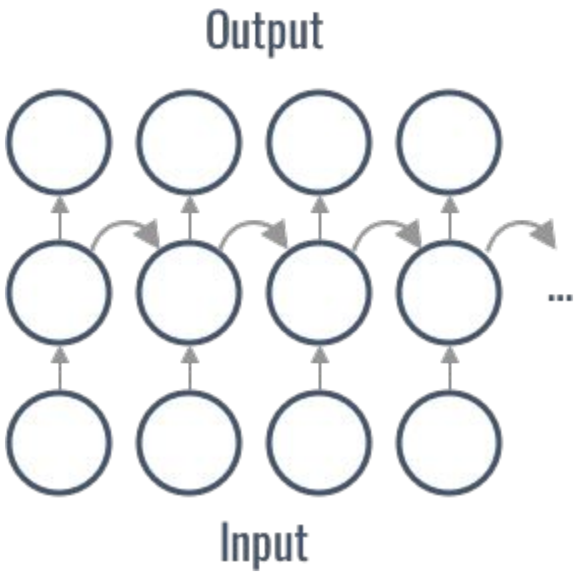
$$r_t^I = 1/c \sum_{i=1}^{c} d_{\cos}(s_t - s_{t-i}, g_{t-i})$$

$$\nabla \pi_t = A_t^D \nabla_\theta \log \pi(a_t | x_t; \theta)$$

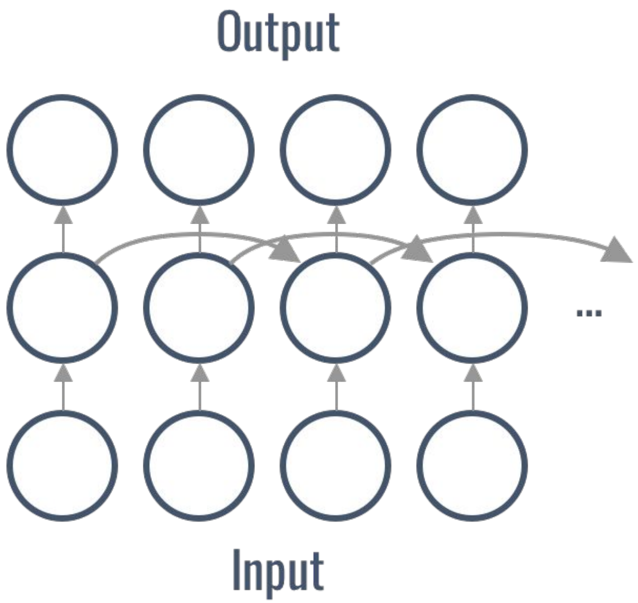
$$A_t^D = (\boxed{R_t + \alpha R_t^I} - V_t^D(x_t; \theta))$$

Manager RNN: Dilated LSTM

- Memories over longer periods
- Outputs are summed over *c* steps
- Performs better

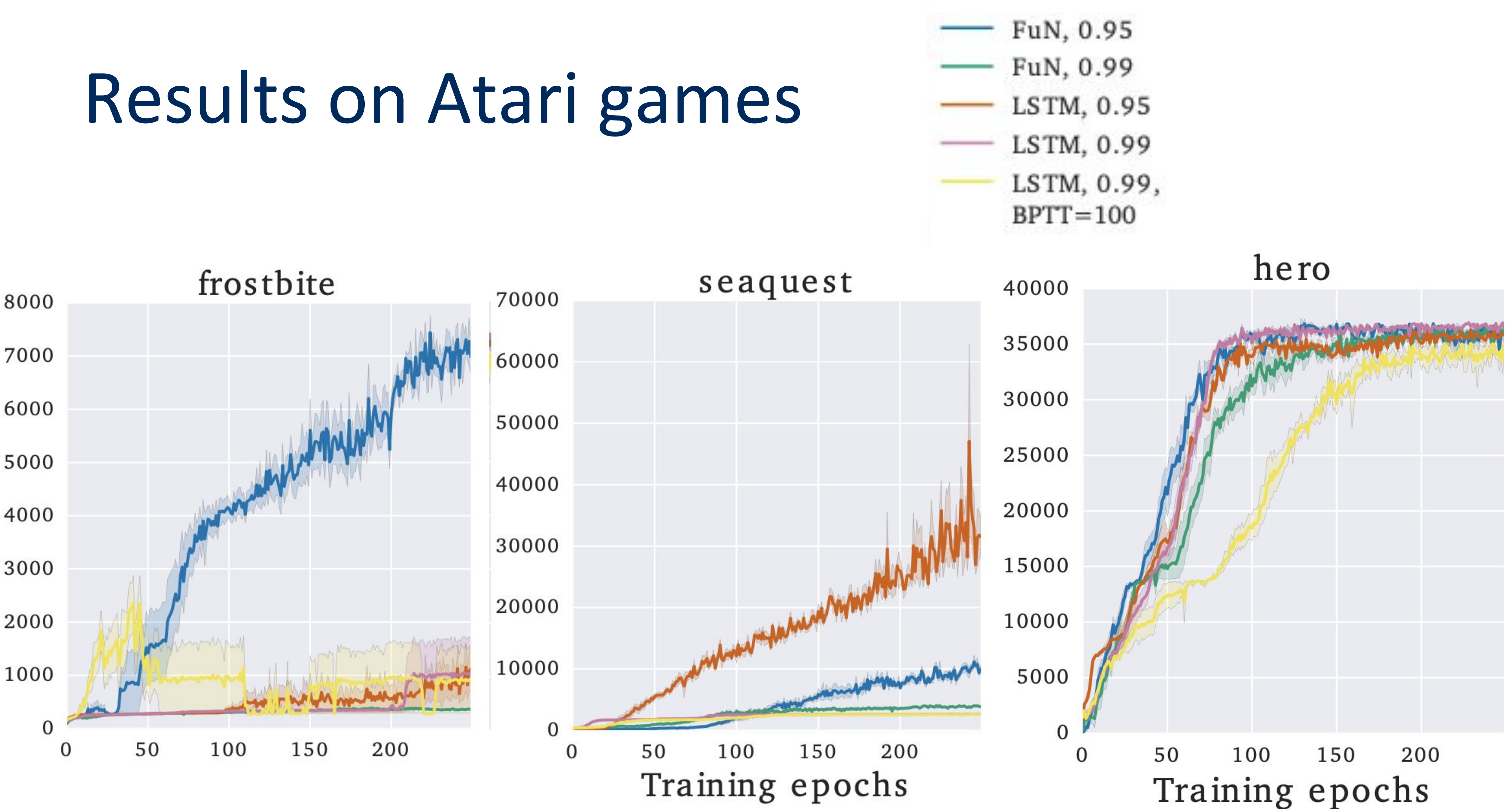


"Standard" RNN

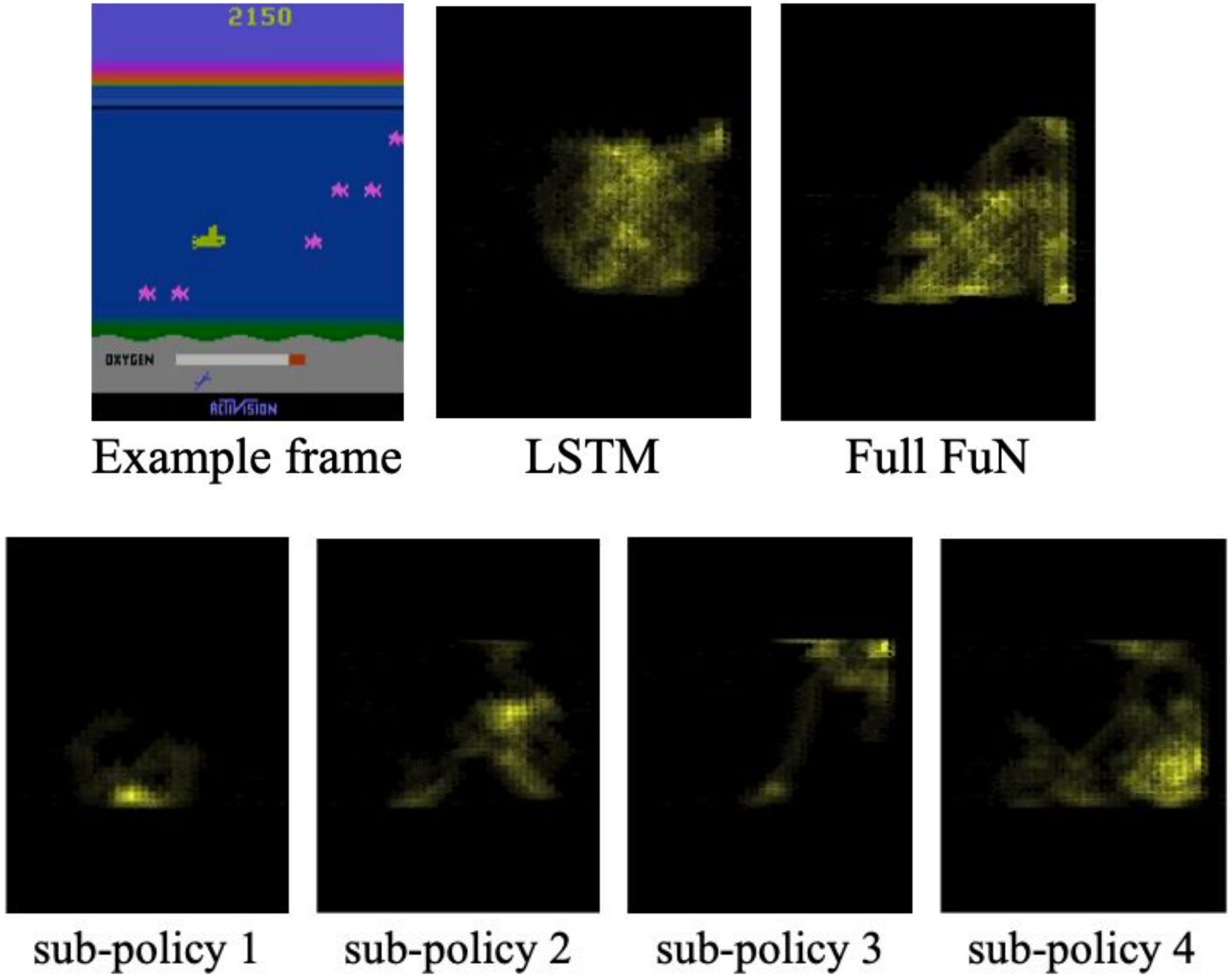


Dilated RNN

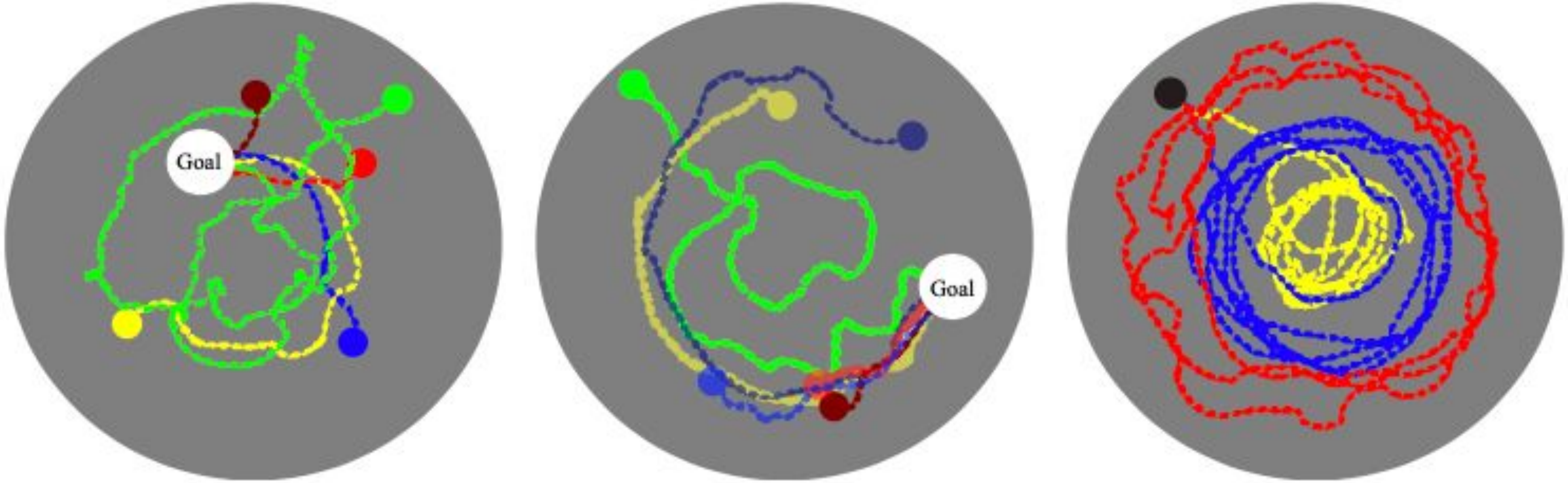
Results on Atari games



Sub-policies inspection

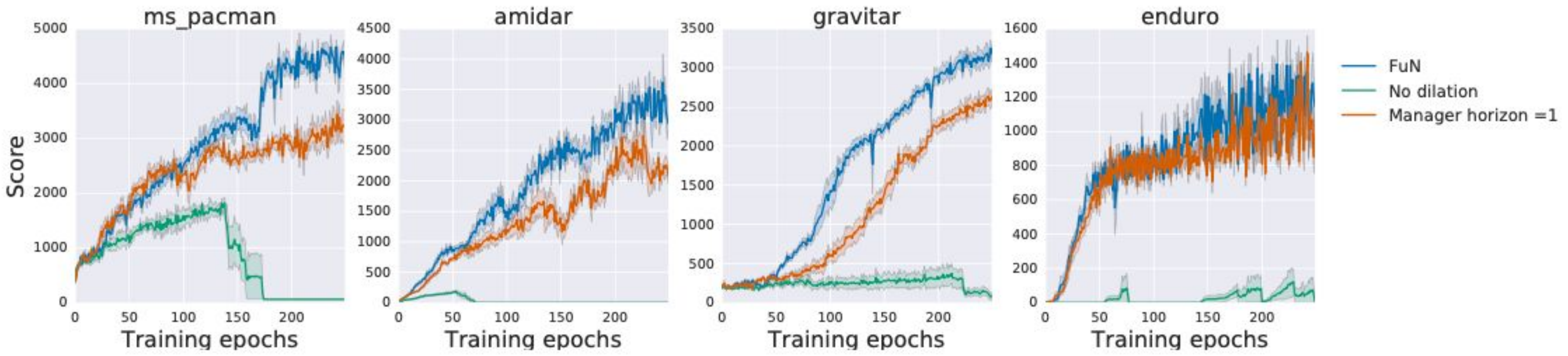


Sub-policies inspection



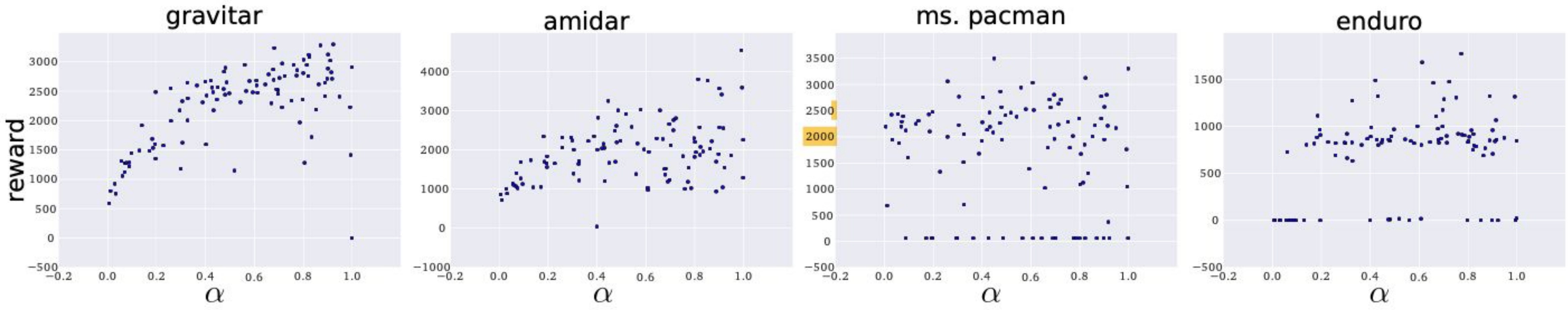
(b)

Is the Dilated LSTM important?



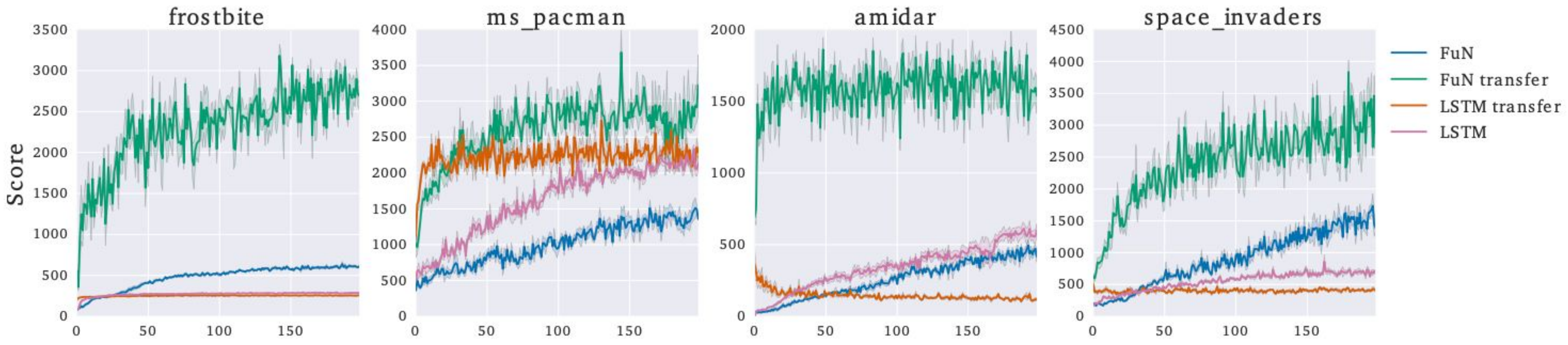
Influence of α

$$R_t + \alpha R_t^I$$

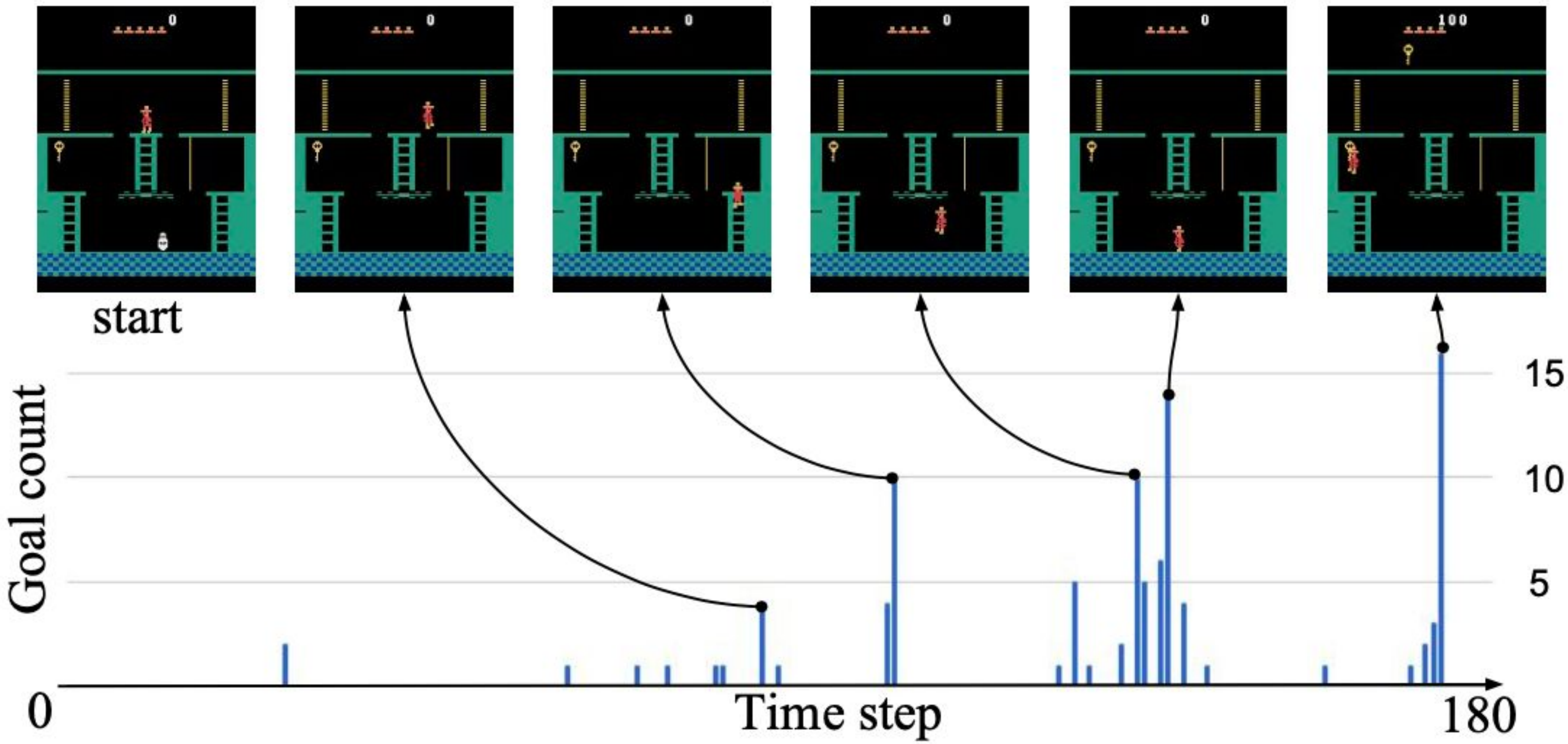


Transfer Learning

- They changed the number of action repeat



Did it solve Montezuma's Revenge?



Sum up of the results

- Using directional goals works well
- Better long-term credit assignment
- Better transfer learning
- Manager's goals corresponds to **different sub-policies**
- Dilated LSTM is essential for good performance
- Meticulous ablation studies - proving their points with evidence (vs claiming SOTA)

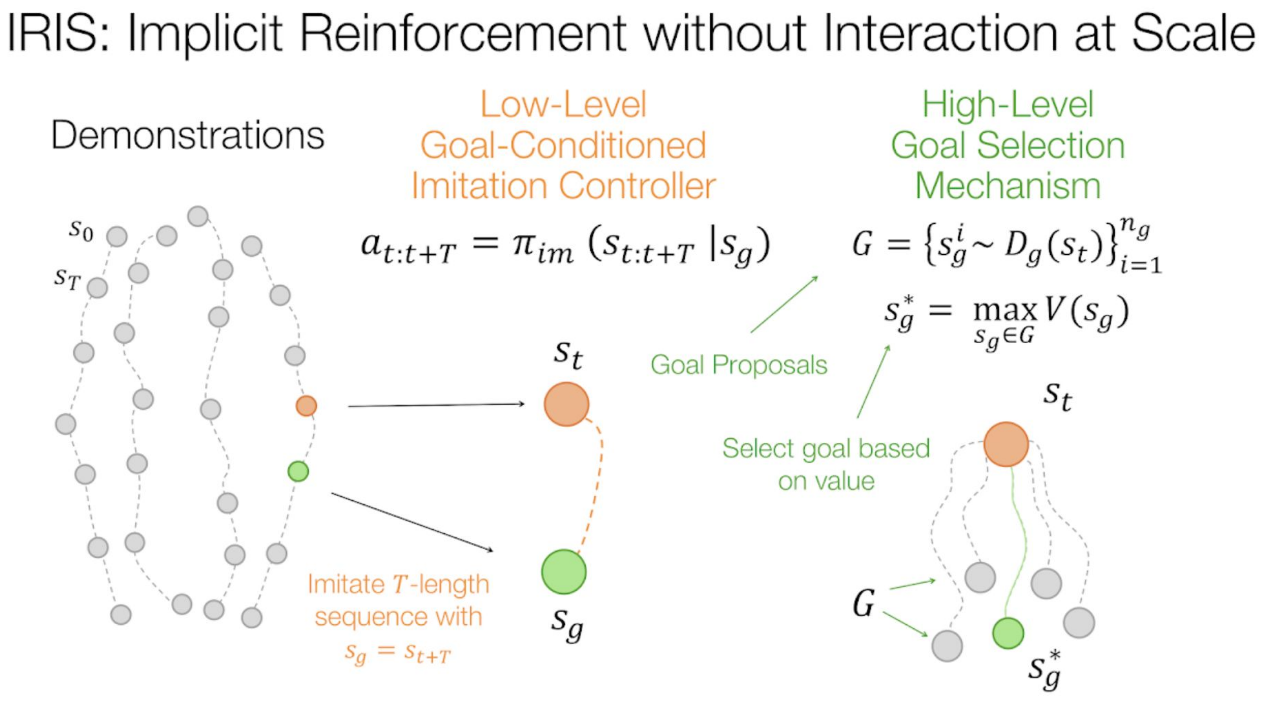
FeUdal Network vs Options Framework

- Only one Worker vs many options
  - Memory efficient
  - Cheaper computationally
- Meaningful goals producing different sub-policies
- “Standard” MDP

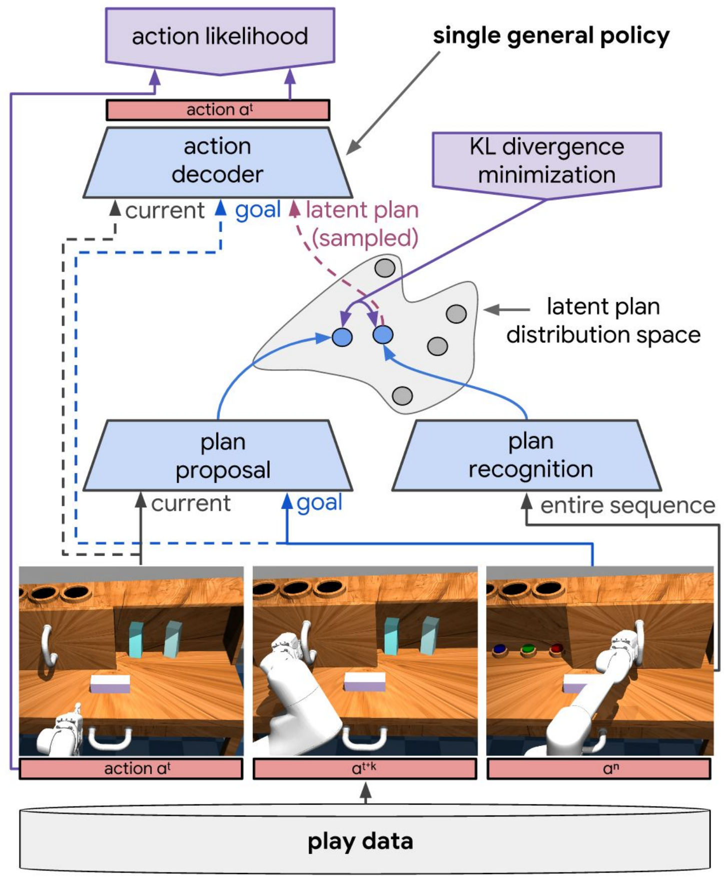
Contributions (recap)

- Differentiable model that implements Feudal RL
- *Approximate transition policy gradient* for training the Manager
- Directional goals instead of absolute
- Dilated LSTM

Has this method inspired others?



https://sites.google.com/stanford.edu/iris/



Learning Latent Plans from Play
https://learning-from-play.github.io/

Open challenges

- Montezuma's revenge remains a challenge
- Maybe using deeper hierarchy and different time scale?
- Transfer learning from an environment to another?