FeUdal Networks for Hierarchical Reinforcement Learning

Presented by Wei Bi

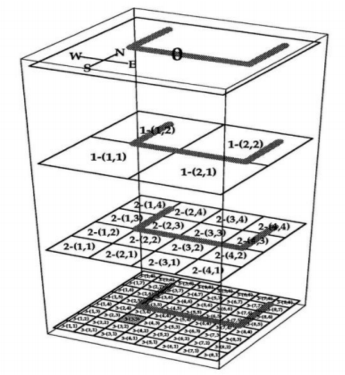
DeepMind, ICML 2017

2018 July 3

Problem with standard RL

- Long term credit assignment
- Sparse reward signals

Original FeUdal Reinforcement Learning



Each action translates into levels of hierarchy within an agent:

- Simple Grid-Environment
- Actions: N,S,E,W and *; * Action lets a lower-level manager search.
- Trained with traditional Q-Learning.

The Proposed FeUdal Networks

- The top level - Manager: set goals at a lower temporal resolution in a latent state-space that is itself learnt;
- The lower level - Worker: operates at a higher temporal resolution and produces primitive actions.

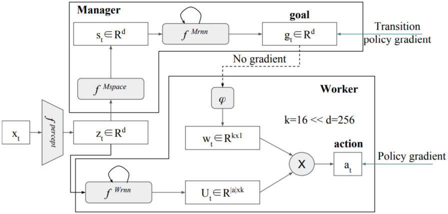


Figure 1. The schematic illustration of FuN (section 3)

Consider a task-oriented dialogue problem (e.g. travel planning):

- The Manager selects the subtask(e.g. book-flight-ticket); But this paper allows a continuous subtask space.
- The Worker takes a sequence of actions with the subtask in control (e.g. departure time, number of tickets etc.)

The Proposed FeUdal Networks: Manager (Forward)

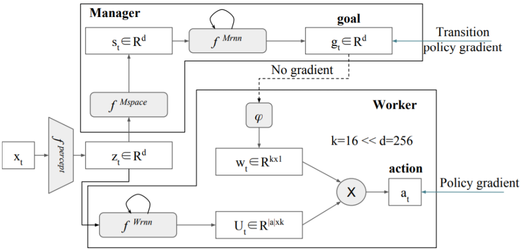


Figure 1. The schematic illustration of FuN (section 3)

- The state x_t is projected into a d-dimensional space Z and we have its embedding vector z_t;
- The manager computes a latent representation s_t which is a "higher-level" embedding of the state;
- The manager then treats s_t and g_t as a sequence and uses a dilated-LSTM to output a goal vector g_t:

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

The Proposed FeUdal Networks: Worker (Forward)

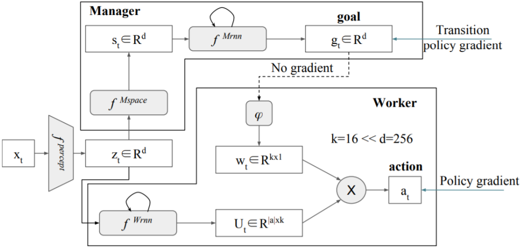


Figure 1. The schematic illustration of FuN (section 3)

- The worker uses a traditional RNN to compute a matrix U_t based on the state embedding z_t: $h_t^W, U_t = f^{Wrnn}(z_t, h_{t-1}^W)$
- U_t can be considered a set of policies, with each row corresponding to an action that the manager can select from.
- The manager takes the goal embeddings from the manager, performs a no-biased linear transform: $w_t = \phi(\sum_{i=t-c}^{t} g_i)$
- w_t is used to weight the policies in U_t: $\pi_t = \text{softmax}(U_t w_t)$.

The Proposed FeUdal Networks (Backwards)

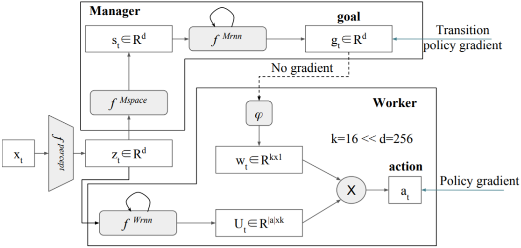


Figure 1. The schematic illustration of FuN (section 3)

- The Manager and the Worker are trained independently.
- The Manager is trained to choose goals with semantic meaning as advantageous directions in the latent space
- The Worker is given intrinsic reward for following the goals set by the manager.

The Proposed FeUdal Networks: Manager (Backwards)

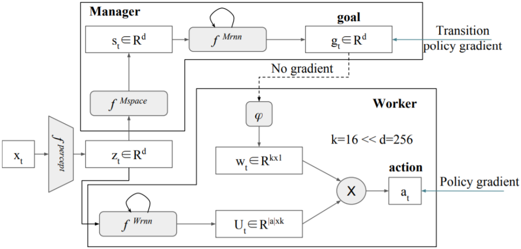


Figure 1. The schematic illustration of FuN (section 3)

- Compute the Manager's advantage function $A_t^M = R_t - V_t^M(x_t, \theta)$, with $V_t^M(x_t, \theta)$ computed using an internal critic.
- Computes the cosine distance at a horizon "c" in the direction of the goal and compute the gradient of the Manager as: $\nabla g_t = A_t^M \nabla_\theta d_{cos}(s_{t+c} - s_t, g_t(\theta))$.
- The Manager is not trained by gradients from the Worker, but from the advantageous directions in the state space S.

Transition Policy Gradients for the Manager

- Assume a high-level policy $o_t = \mu(s_t, \theta)$ that selects among sub-policies (possibly from a continuous set), which are fixed duration behaviours (lasting for c steps).
- Model a transition policy: $\pi^{TP}(s_{t+c}|s_t) = p(s_{t+c}|s_t, o_t)$, with $p(s_{t+c}|s_t, o_t) \propto e^{d_{cos}(s_{t+c}-s_t, g_t)}$.
- The gradients with respect to the policy parameters:

$$\nabla \pi_t^{TP} = A_t^M \nabla_\theta \log p(s_{t+c}|s_t, \theta).$$

The Proposed FeUdal Networks: Worker (Backwards)

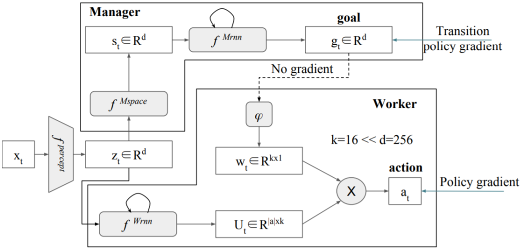


Figure 1. The schematic illustration of FuN (section 3)

- Similar to the traditional policy update with the use of an intrinsic reward

$$A_t^D = R_t + R_t^I - V_t^D(x_t, \theta)$$

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

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

Experiments on ATARI game - Montezuma's revenge

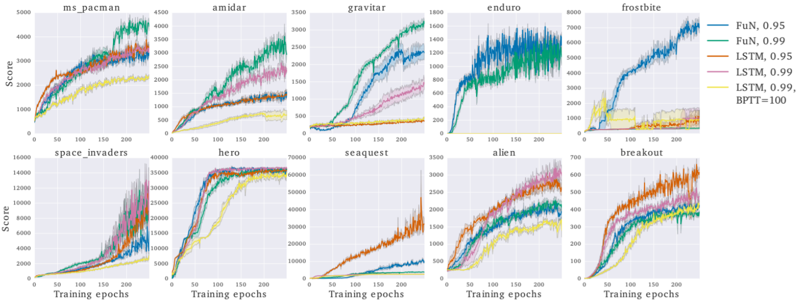


Figure 4. ATARI training curves. Epochs corresponds to a million training steps of an agent. The value is the average per episode score of top 5 agents, according to the final score. We used two different discount factors 0.95 and 0.99.

Conclusions

- Can be readily to replace flat RL in decoding.
- How to define the goal of manager?
  - Just let it be a latent variable - CVPR2018
  - The subgoal of task-oriented dialogue - EMNLP2017
  - Can we define a better goal with meaningful interpretations in chichat-setting?
- Instead of using it in the decoder, can we apply HRL in the memory construction or anywhere currently RL can be used in text generation?