Reinforcement Learning & Text Games

Hadi Vafaei

0) Problem and motivation

- Reinforcement Learning has been successful in playing Atari games; however, not as much progress has been made for text-based games due to additional language challenges
- Our goal is to build RL agents that can play text-based games. For this study we chose to use TextWorld. Example screenshot from the game →
- At each stage of the game:
  - the agent is provided with a text description of the world state (obs)
  - agent plays by generating text commands based on these observations
  - sometimes useful commands lead to rewards
- These reward signals can be used to train the agent
- The goal is to accumulate high rewards in little time

1) Challenges

- Challenges on the language side include:
  - Natural language understanding
  - Text generation
  - Dealing with combinatorial action space

We provide the agent with the list of admissible commands to circumvent some of these difficulties for now…

- RL challenges include:
  - Sparsity of reward
  - Credit assignment
  - Exploration/exploitation

2) Technical Approach

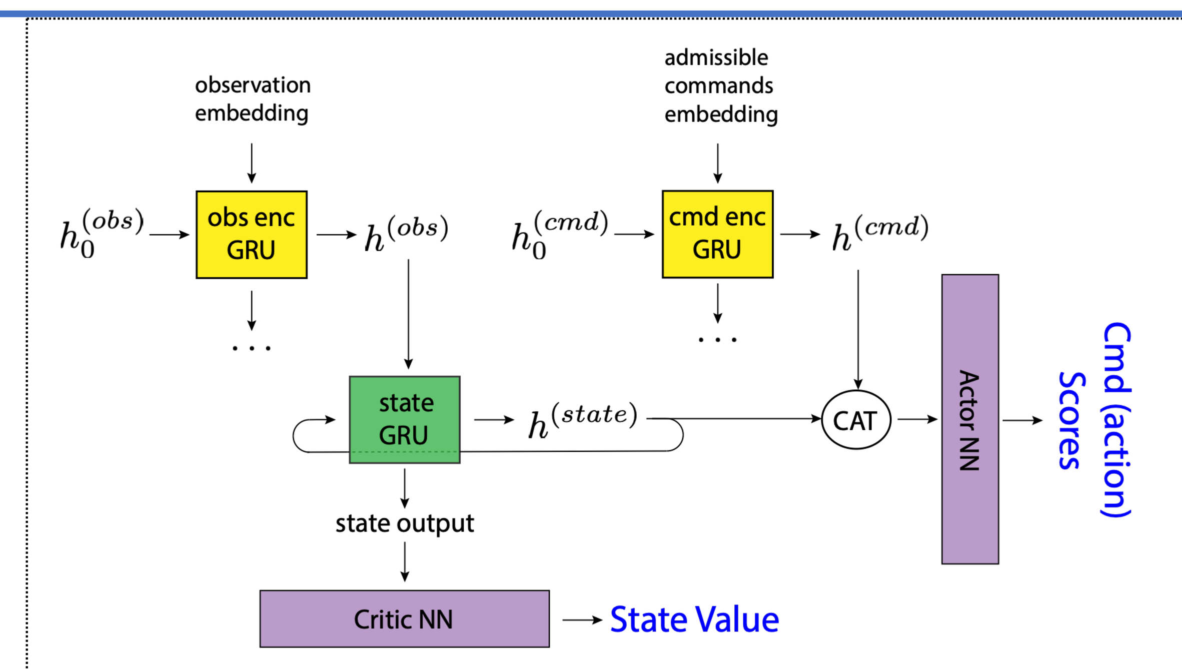
- GRUs + Advantage Actor Critic (A2C): a policy gradient based learning algorithm

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\; v_t\right] && \text{REINFORCE}\\
&= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\; Q^w(s,a)\right] && \text{Q Actor-Critic}\\
&= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\; A^w(s,a)\right] && \boxed{\text{Advantage Actor-Critic}} \Rightarrow \text{This is what we used}\\
&= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\; \delta\right] && \text{TD Actor-Critic}
\end{aligned}
$$

taken from CMU CS 10703 lecture slides

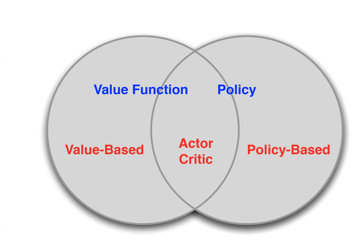
- Inspired by Soft Actor Critic algorithm (SAC), we experimented with adding an entropy term to our loss to see if it helps (it doesn't)
- Advantage function is defined as:

$$A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$$



- Discrete actions: Softmax Policy

- Value Based
  - Learnt Value Function
  - Implicit policy (e.g. ϵ-greedy)
- Policy Based
  - No Value Function
  - Learnt Policy
- Actor-Critic
  - Learnt Value Function
  - Learnt Policy

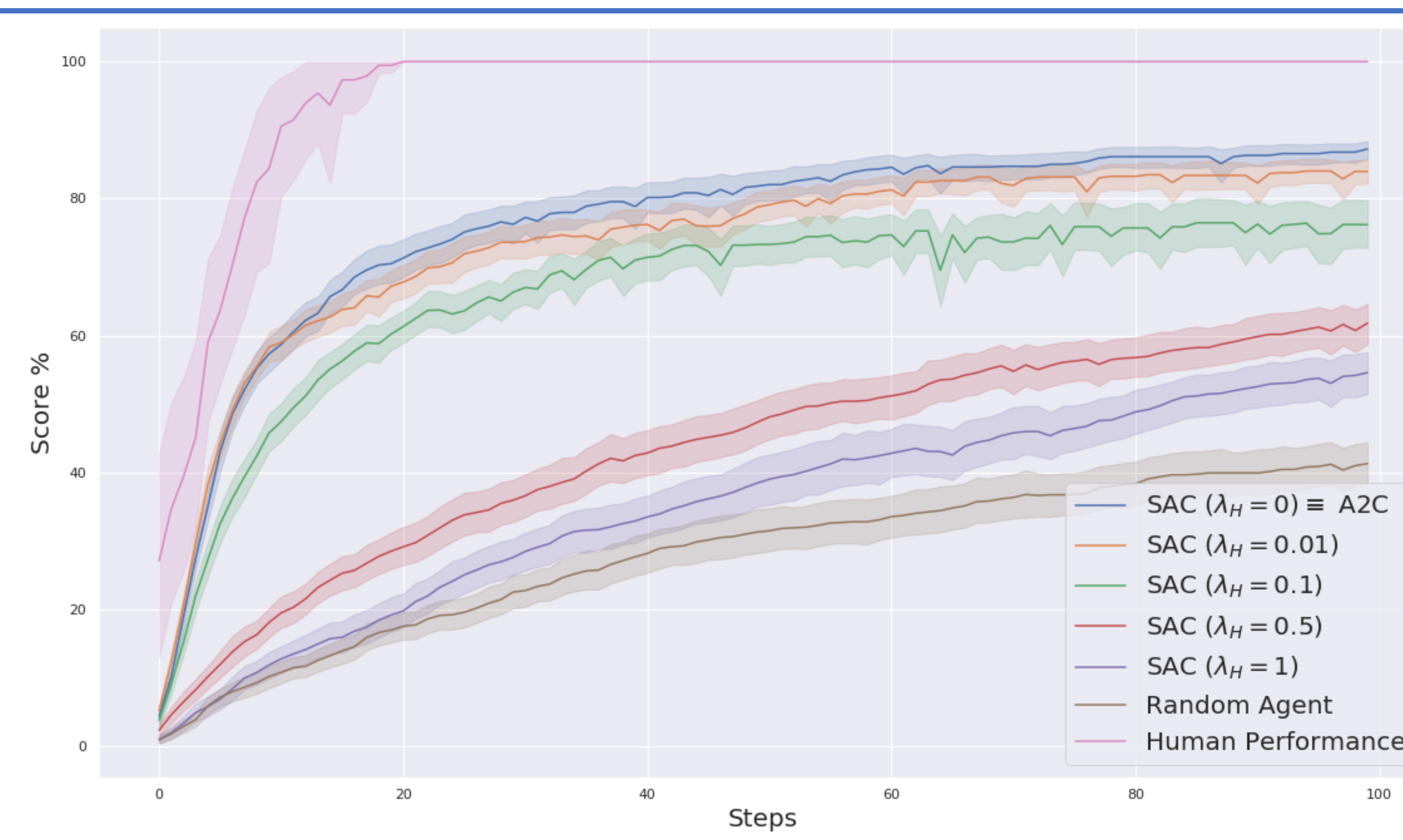


From David Silver's RL course

3) Results

Experiment # 1 (simple games):

- Trained on 100 games with dense rewards and a detailed description of goal
- Validated on 20 games from similar distribution
- Observations:
  - Our agent consistently outperforms the random baseline by a large margin but is inferior to humans
  - Including the entropy term hurts model performance. Perhaps this is because SAC is better suited for tasks with continuous action space

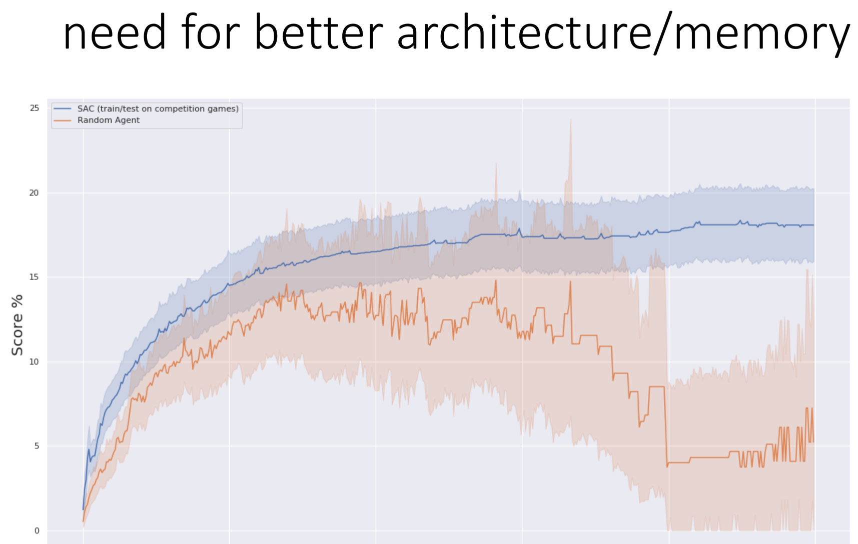


Here lambda is the entropy term hyperparameter in the loss:

$$Loss = policy\ loss + value\ loss + \lambda_H * H$$

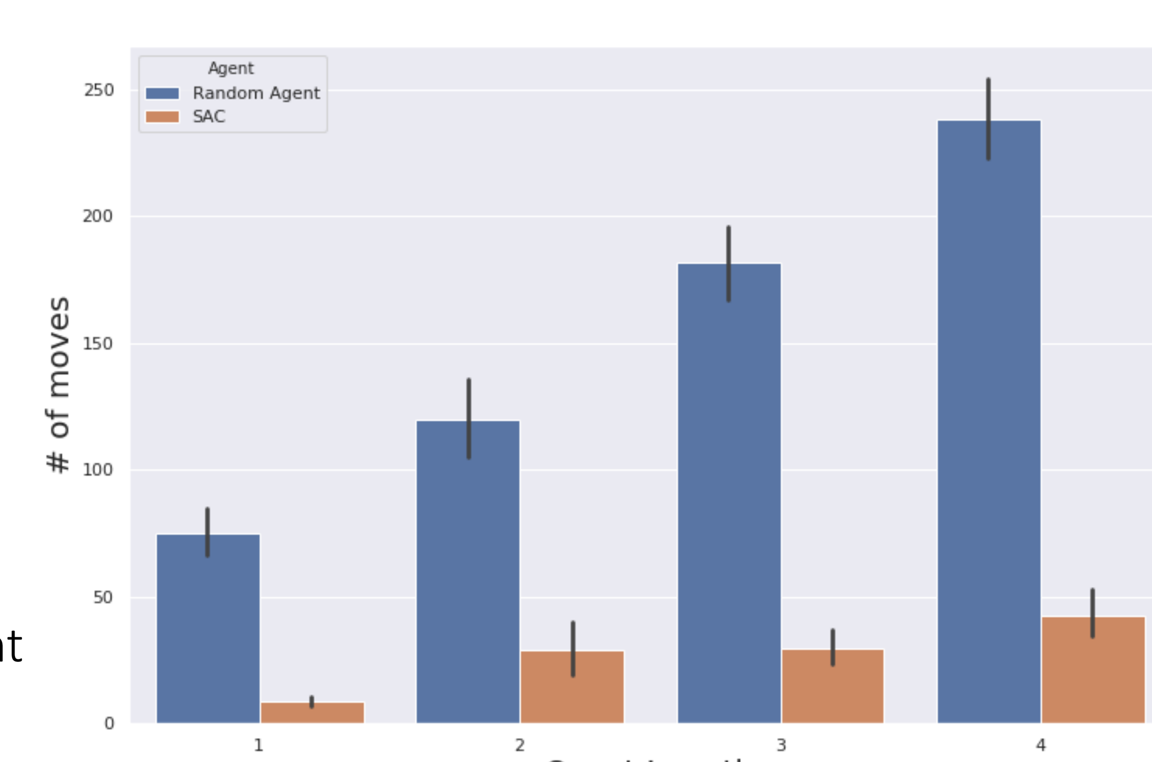
Experiment # 2 (hard games):

- We trained the same model on 4440 games from the competition
- Validated on 222 games
- Due to difficulty of the tasks, our
- agent barely outperforms the random baseline. This hints at the need for better architecture/memory



Experiment # 3 (quest length):

- Trained/validated on games with quest length of {1, 2, 3, 4}
- For example, quest length of 1 means the agent instantly wins the game at the first stage if it makes the right choice
- Observations:
  - Performance much better than the random agent
  - These games are simple, so humans will win a game of quest length N in roughly ~N steps
  - Therefore, our agent is again inferior to humans



4) Future Work

- Meta-Learning: better than joint training on batch of games?
- Some sort of supervision: Imitation Learning?
- Bette model architectures, structured memory

Key References:

- TextWorld: A Learning Environment for Text-based Games [Côté et. al. 2018]
- Language Understanding for Text-based Games using Deep Reinforcement Learning [Narasimhan et. Al. 2015]
- Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor [Haarnoja et. Al. 2018]